January 1981
USA $2.95/DM 9.80

kilobaud

MICROCOMPUTING T.M.

(DESK-TOP COMPUTERS)

Assemble a Super Business System

Probe 16-Bit Technology

Computervans and Software Stores □ Enhancing H8 BASIC □ Apple Connections □ OSI Reverse Video □ TRS-80 in the Darkroom □ 6800 Accounting System □ Implementing File Systems □ PET Spectrum Analyzer

A Few Extraordinary Products for Your 6800/6809 Computer

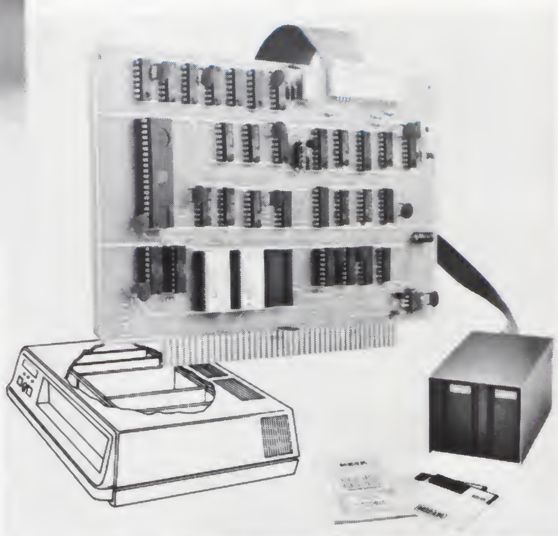
From Percom . . .

SS-50 Bus LFD-400™ and LFD-800™ Systems

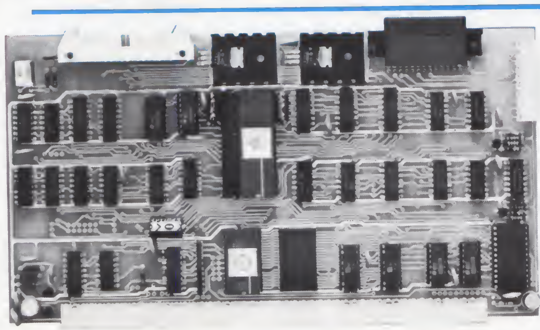
Low Cost Mini-Disk Storage in the Size You Want 14

Percom mini-disk systems start as low as $599.95, ready to plug in and run. You can't get better quality or a broader selection of disk software from any other microcomputer disk system manufacturer — at any price!

Features: 1-, 2- and 3-drive systems in 40- and 77-track versions store 102K- to 591K-bytes of random access data on-line • controllers include explicit clock/data separation circuit, motor inactivity time-out circuit, buffered control lines and other mature design concepts • ROM DOS included with SS-50 bus version — optional DOSs for EXORciser* bus • extra PROM sockets on-board • EXORciser* bus version has 1K-byte RAM • supported by extended disk operating systems; assemblers and other program development/debugging aids; BASIC, FORTRAN, Pascal and SPL/M languages; and, business application programs.



EXORciser Bus LFD-400EX™ -800EX™ Systems*



Versatile Mother Board, Full-Feature Prototyping Boards 15

Printed wiring is easily soldered tin-lead plating. Substrates are glass-epoxy. Prototyping cards provide for power regulators and distributed capacitor bypassing, accommodate 14-, 16-, 24- and 40-pin DIP sockets. Prototyping boards include bus connectors, other connectors and sockets are optional.

MOTHERBOARD — accommodates five SS-50 bus cards, and may itself be plugged into an SS-50 bus. Features wide-trace conductors. Price: $21.95

SS-50 BUS CARD — accommodates 34- and 50-pin ribbon connectors on top edge, 10-pin Molex connector on side edge. Price: $24.95.

SS-30 BUS CARD — 1¼-inch higher than SWTP I/O card, accommodates 34-pin ribbon connector and 12-pin Molex connector on top edge. Price: $14.95.

The SBC/9™. A "10" By Any Measure. 13

The Percom SBC/9™ is an SS-50 bus compatible, stand-alone Single-Board Computer. Configured for the 6809 microprocessor, the SBC/9™ also accommodates a 6802 without any modification. You can have state-of-the-art capability of the '09. Or put to work the enormous selection of 6800-coded programs that run on the '02.

The SBC/9™ includes PSYMON™, an easily extended 1-Kbyte ROM OS. Other features include:

- Total compatibility with the SS-50 bus. Requires no changes to the motherboard, memory or I/O.
- Serial port includes bit-rate generator. RS-232-C compatible with optional subminiature 'D' connector installed. 10-pin Molex connector provided.
- Eight-bit, non-latched, bidirectional parallel port is multi-address extension of system bus. Spans a 30-address field; accommodates an exceptional variety of peripheral devices. Connector is optional.
- Includes 1-Kbyte of static RAM.
- Costs only $199.95 with PSYMON™ and comprehensive users manual that includes source listing of PSYMON™.

™ trademark of Percom Data Company, Inc.
* trademark of the Motorola Corporation.
Prices and specifications subject to change without notice.

The Electric Window™: Instant, Real-Time Video Display Control 16

Memory residency and outstanding software control of display format and characters make this SS-50 bus VDC card an exceptional value at only $249.95. Other features:

- Generates 128 characters including all ASCII displayable characters plus selected Greek letters and other special symbols.
- Well-formed, easy-to-read 7x12-dot characters. True baseline descenders.
- Character-store (display) memory included on card.
- Provision for optional character generator EPROM for user defined symbols.
- Comprehensive users manual includes source listing of Driver software. Driver — called WINDEX™ — is also available on mini-diskette through the Percom Users Group.

PERCOM

PERCOM DATA COMPANY, INC.
211 N. KIRBY GARLAND, TEXAS 75042
(214) 272-3421

Products are available at Percom dealers nationwide. Call toll-free, 1-800-527-1592, for the address of your nearest dealer, or to order direct.

INTRODUCING THE EMULATOR™

The trouble with video terminals today is that most of the low-cost models just don't have the performance to handle your tough applications. And the few that do are usually not compatible with your existing system. But now, Intertec has resolved this age old dilemma with the introduction of its new Emulator™ Video Terminal.

The $895* Emulator™ performs exactly as you command. With the depression of just a few keys, Emulator users can select terminal control codes of any one of four popular video terminals. The Lear-Siegler ADM-3A. The Soroc 1Q-120. The DEC VT-52. Or the Hazeltine 1500. Incredible! It's like having four terminals for the price of one.

But, best of all, not only does the Emulator replace these terminals, it outperforms them by offering enhanced user-oriented features. Features that those other terminals just don't have - at any price.

Standard Emulator™ features include: a sharp, crisp 12" non-glare screen with a full 24 line by 80 column display. Twin RS232C serial ports - one for the host computer and one for your printer. Four separate cursor control keys. A separate 18 key numeric pad. Keyboard selectable baud rates and operating modes. And, a host of visual attributes.

No matter which dumb or smart terminal you're using today, don't buy another until you check out our new Emulator™. You'll get the performance of four terminals for the price of one. And you'll probably save hundreds of dollars over the price you paid for your last terminal. Plus, you'll get unparalleled reliability, nationwide service and quick delivery. Call or write us today for all the details. Intertec terminals are distributed worldwide and may be available in your area now.

INTERTEC DATA SYSTEMS®

2300 Broad River Rd, Columbia, SC 29210
(803) 798-9100 TWX: 810-666-2115

343

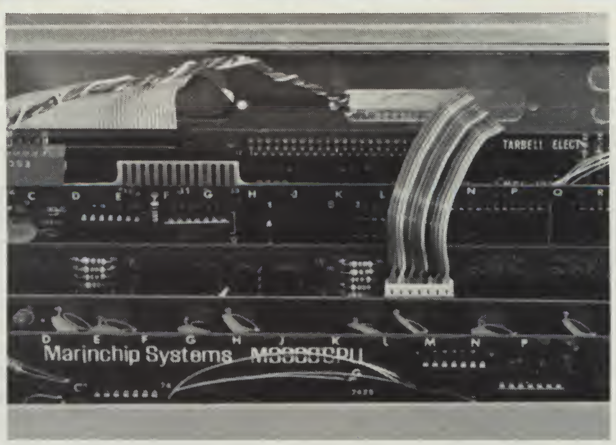
*Quantity one - Dealer inquiries invited

MICROCOMPUTING

PUBLISHER/EDITOR
Wayne Green
EXECUTIVE VICE PRESIDENT
Sherry Smythe
EDITORIAL MANAGER
Jeff DeTray
PUBLICATIONS MANAGER
Edward Ferman
MANAGING EDITOR
Dennis Brisson
ASST. MANAGING EDITOR
Susan Gross
COPY EDITOR
Eric Maloney
EDITORIAL ASSISTANT
Linda Stephenson
ADMINISTRATIVE ASSISTANTS
Cresca Clyne, Pat Graham, Nancy Noyd
ASSOCIATE EDITORS
Robert Baker, Ken Barbier, Frank Derfler, Jr., Rod Hallen, Peter Stark, Sherm Wantz

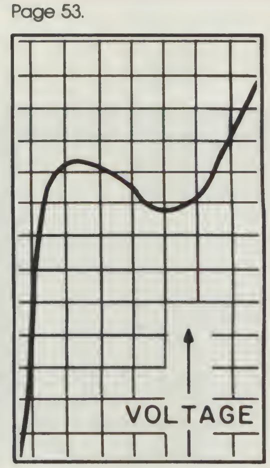
MANUFACTURING MANAGER
Noel Self
PRODUCTION MANAGER/PUBLICATIONS
Nancy Salmon
ASST. PRODUCTION MANAGER
Michael Murphy
ART DIRECTOR
Diana Shonk
PRODUCTION DEPARTMENT
Joan Ahern, William Anderson, Jr., Steve Baldwin, Linda Drew, Robert Drew, Bob Dukette, Bruce Hedin, Kenneth Jackson, Ross Kenyon, Maryann Metivier, Theresa Ostebo, Jane Preston, Dion Owens, Patrice Scribner, Susan Symonds, Thomas Villeneuve
PHOTOGRAPHY
William Heydolph, Terrie Anderson, Bill Suttenfield
TYPESETTING
Barbara Latti, Sara Bedell, Michele DesRochers, Luann Keddy, Mary Kinzel, Linda Locke, Karen Podzycki

CORPORATE CONTROLLER
Charles Garniss, Jr
PLANT MANAGER
Leatrice O'Neil
ACCOUNTING MANAGER
Knud Keller
CIRCULATION MANAGER
Debra Boudrieau
CIRCULATION
Doris Day, Pauline Johnstone
BULK SALES MANAGER
Ginnie Boudrieau
PUBLIC RELATIONS
Joseph Wilson
ADVERTISING
603-924-7138
Kevin Rushalko, Mgr.
Marcia Stone, Hal Stephens



Page 29.

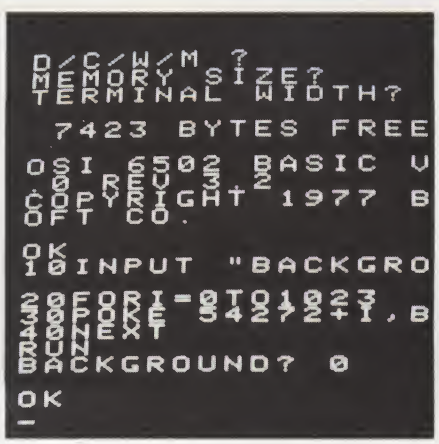
Page 53.



Contents: January 1981

Volume V No. 1

Page 106.



Page 176.

This month:

In the four years since the inception of **Kilobaud Microcomputing**, the nature of the microcomputer magazine audience has changed significantly. The hardware-oriented hobbyist of 1977 has been superseded by the businessman, educator and scientist of 1981. Also, as the hardware has become increasingly reliable, there has been a growing realization that software is the key to the future growth of the industry.

We have begun to address this new audience with articles and special issues devoted to their needs. Business/professional articles are flagged with a unique identifier. Last June saw an issue devoted to the educational uses of microcomputers. Issues emphasizing word processing and data communications are among those planned for this year. Our shift in direction is a gradual one (indeed, we will always have a place in the magazine for the hardware tinkerer and homebrew hacker, who has been with us from the beginning and remains with us).

As we begin our fifth year of operation, the magazine will take on a new look in format, as well as content. New type styles, layouts and design ideas have been introduced to bring you a more readable and enjoyable magazine. And to keep you abreast of the happenings in this rapidly changing field, we have introduced several new features, which will appear regularly each month, and have beefed up our regular columns.

We hope you find that these changes (some are subtle; some are very blunt) live up to the quality you have come to expect from this magazine in just its few short years. As always, any ideas, suggestions or constructive criticisms are welcome. Let's keep the lines open.

–The Editors

This month's cover:

The Marinchip Systems M9900 CPU board, which forms the basis of our 16-bit "Super Business System" cover story (p. 29). Photo by Martin Paul.

Kilobaud *Microcomputing* (ISSN 0192-4575) is published monthly by Wayne Green, Inc., 80 Pine St., Peterborough NH 03458. Subscription rates in U.S. are $25 for one year and $53 for three years. In Canada: $27 for one year only, U.S. funds. Foreign subscriptions (surface mail)—$35 for one year only, U.S. funds. Foreign air mail subscriptions —$62 for one year only, U.S. funds. Canadian Distributor: Micron Distributing, 409 Queen St. West, Toronto, Ontario, Canada M5V 2A5. In Europe, contact: Monika Nedela, Markstr. 3, D-7778 Markdorf, W. Germany. South African Distributor: KB Microcomputing, PO Box 782815, Sandton, South Africa 2146. Australian Distributor: Electronic Concepts, Attention: Rudi Hoess, 55 Clarence Street, Sidney 2000, Australia. Second-class postage paid at Peterborough NH 03458 and at additional mailing offices. Phone: 603-924-3873.

Wayne Green

East Headed West

The Asian Threat

The American manufacturers of microcomputer equipment have pretty much had it their own way ever since the industry got started in 1975. Even in Europe the American products have been the best sellers. Up until fairly recently this was also true in Japan and most of Asia—but that is changing.

The money and pressures for miniaturization in the space race of a few years back brought on our development of microelectronics. The microprocessor was the result of this evolution, making possible the microcomputer. From there, the demands of our 200-million-strong population and its relatively high, displosive income fanned the flames, bringing a myriad of poorly organized firms into business to satisfy this market.

The Japanese, not being entrants in the space race, let us spend our billions of dollars developing LSI chips, then they stepped in. Unlike the U.S., which has had virtually no increase in the number of electronic engineer graduates in the last 17 years, Japan has had a substantial growth, and they are far ahead of us in the number of both engineers and technicians. This shows when it comes to developing consumer products.

While more and more of the Japanese-designed products are being made in such places as Korea, Taiwan, Hong Kong, Singapore and the Philippines, the new technology is most definitely coming from Japan. In the microcomputer field this means that while Japan got started a couple years late, now they are moving ahead like a steamroller, and many smaller U.S. firms are in a position to get crushed.

A year ago, when I visited Japan, one of the best-selling microcomputers was the Apple. On my latest trip I rarely saw one, and was told by several of the larger dealers that Apple sales in Japan are no longer significant due to the competition from new Japanese systems.

In America, the major well-known firm making systems is Radio Shack—and perhaps Texas Instruments could be counted in there, though they do not yet seem to have found a way to really market their product. In Japan everyone is in the act. There are excellent micros being marketed by Casio, Hitachi, Sanyo, Seiko, NEC, Sharp, Toshiba and so on. The scary part for us is that most of these firms have their eye on the fast-growing and almost unlimited American market.

Already announced as entrants in the U.S. are Panasonic, Quasar, Casio, NEC, Sord and Sharp. I have good reason to believe that Toshiba, Hitachi, Sanyo and others will be here shortly, once they figure out how best to market their systems in the U.S.

Can the Japanese design better microcomputers than we? Since they are head and shoulders above us on the design of most electronic equipment today, the reasonable answer is yes. In looking over the latest electronic equipment, I am hard put to find American products which are as innovative as those from Japan. Just looking around at the gadgets I've picked up recently gives me a good indication. I try to keep abreast of consumer electronics technology and familiarize myself with new developments.

On my wrist is a Casio C-80 calculator watch. It is so good, in my estimation, and so incredibly inexpensive, that I wonder if any other watch firms are going to survive. I do have a second watch—for skin diving—and it's another new, inexpensive Casio. On my desk are a Sony (M-400B) microrecorder, which has paid for itself several times over already; a new Sony TCS-300 stereo minirecorder, which is invaluable for recording talks and meetings; a Casio Melody-90 calculator; and a Sharp Memowriter (EL-7000). In my desk drawers are older calculators such as the Casio ML-80, MQ-1, MQ-2 and CQ-1. I keep a tiny Sanyo portable TV handy, and Sherry is hardly ever without her Sharp Pocket Computer and microrecorder for programs and data.

Until such time as we figure out how to attract more people to engineering and technician work, I think we are going to have to try to accept Japan leading us in electronic technology—certainly on a consumer electronics products level. I think that we are going to see an increasing number of Japanese microcomputers coming into our country, and, unless something goes wrong with their system (which hasn't failed them recently), they will be able to gradually push many of the American firms out of business.

I should add a caveat. The above takes for granted that neither the U.S. nor the Japanese firms will decide to do anything serious about software. Since no U.S. firm has as yet, nor have I seen any glimmer of hope that any will, it is possible that the Japanese will be as blind to this aspect of selling computers as the U.S. firms. If so, the hardware will sell largely on the basis of its excellent design. Unless our experiences in ham gear, in hi-fi equipment, in auto sound, in calculators, in video cameras, in video recorders, in cassette records, in CB equipment, and on down the line are unique, they have a good chance to take over still another of our high-technology industries. Did I forget to mention TV sets and transistor radios?

Certainly, one good look at the current crop of Japanese microcomputers is not likely to instill confidence in our American products. Well, you may ask, what about the new Radio Shack TRS-80PC? That's certainly innovative and indicative of what *can* be done. Right, and it comes from Sharp in Japan. Sherry bought one in Tokyo, and it *is* a darned good product. Perhaps the American firms will be able to cope with the Japanese talents by merely doing the American marketing of the products.

Speaking of the PC, I'd like to make sure that readers who have one of those or the Sharp version are well aware that I'm looking for articles and programs on it. That big fat book of programs they sell with the computer is a rip-off. What a bunch of useless programs! One of the

first things we need is a way to interface that contraption to some more memory. The Quasar prototype, which was shown at CES in Chicago last summer, looked incredibly like the Sharp unit, and it had extra memory available to plug in. It also had a modem and even a small printer. I think that the Sharp Memowriter might interface with the Sharp computer to provide a printout. Any experimenters out there?

Unless something changes in the area of software support for one of the American systems, I'd say that events should favor a gradual take-over by the Japanese. This means that stock in American microcomputer firms could turn into a time bomb, with the Japanese lighting the fuse. Should any of the Japanese firms solve the software problem before the U.S. firms, their entry could be made substantially simpler and millions of dollars less expensive.

Another event is happening in Japan which could have long-range effects on our microcomputer industry: the Japanese advances in video disk technology. Some of these techniques are remarkably similar to computer disk technology, and again Japan is way out in front. This could raise hell with our disk firms, should some of the laser writing ideas pan out.

Japan's edge over us has nothing to do with cheap foreign labor at all, for the Japanese auto workers make about what our average wage earners do. The problem for Detroit is that the auto union workers earn double the average American wage, and all of us have had to subsidize this when buying American cars.

Oh, many of the Japanese plants are far more efficient than ours, but this has more to do with our union restrictions than further blindness on the part of auto makers. The Asians figured out some time ago that they would not be able to continue to compete with us on the basis of low wages forever, so they have automated in every possible way to keep down the cost of production.

In Korea, where wage scales are still quite low, I recently visited a color television set factory and saw one of the most modern automated electronic plants I've ever seen. The hand labor there is absolutely minimized, contributing less than $2.50 to the cost of the set. Even if their wages double, they will still be very competitive with any other country.

In addition to the above-average wages for U.S. electronic workers, astronomical wage scales are enjoyed by our few technicians and engineers. We will find ourselves paying the price in the long run.

The prosperity in Silicon Valley was fed by open-ended government contracts in which firms hired people for almost any going wage and used them for a specific contract. Once the contracts were completed, the people would move across the street at an even higher wage to help the next firm with a contract. This encouraged the escalation of salaries all out of proportion to the benefit of the companies designing products—all fed by unlimited government funds. It also encouraged engineers and technicians to jump from one firm to another, chasing ever higher wages.

The Asians don't work like this. Their wages are much more modest, and they stick with a company through thick and thin. This results in far lower costs of training and retraining workers, and the bottom line is the better products which result and the far lower cost of their development.

Thus, in the long run, it seems likely that unless there are some major changes in the way our firms do things, ever-increasing infusions of capital will be needed to keep them afloat in competition with the Japanese. . . and then with the Koreans, Taiwanese, and so forth. We have already seen a couple of systems—the APF and the Video Genie (also known as the TRZ-80 and now the PMC-80)—popping up from Hong Kong. The indications are that Hong Kong has again leaned heavily on foreign R&D, getting into business primarily on the basis of low-cost manufacture.

If the future looks bleak for the smaller makers of microcomputers as a result of the invasion from Asia, what about our two or three major manufacturers? Surely they are in a reasonably impenetrable situation. They have a good head start, they have their sales and service organizations set up and running, they have two or three years of advertising and a whole lot of customers for a customer base. It does look at first glance as though these firms may be on safe footing.

But we've seen the mightiest of them fall in the past. Mits seemed in a position which was really unchallengeable. They were followed by a string of other firms, each appearing to be leaders and on solid ground. Will it be possible for the Japanese to cut into Radio Shack, with their 7000 stores, just on the basis of improved technology and advertising? Well, we'll see.

Instant Software in Asia?

In addition to tending to substantially improved distribution and sales in Australia and New Zealand, plus growing markets in Singapore and Hong Kong, I had to do some follow-up work on our incursion into Korea and Japan. There is now a TRS-80 Korean ROM, so we will be getting to work on translations of our programs into that language. There is a possibility that we'll be organizing a Korean language version of *Microcomputing* too.

My few days in Japan were packed with meetings with firms interested in working on a trading partnership basis, so the prospects for growth there are excellent. This will be important for the long-range future of Instant Software, because we want to be able to draw upon the work of free-lance programmers anywhere in the world and distribute their programs everywhere, in many languages and adapted to whatever computer systems are viable.

With the U.S. programmers a year or two ahead of the Japanese, we have a decided advantage in producing programs for use in Japan. There are, of course, some cultural and business differences, but for the most part the programs of value in the U.S. are of value in Germany, South Africa and Japan.

There will be another tour of the Asian consumer electronics shows in 1981, so perhaps you can get away this time. I don't know if I'll have the time to spare; I just have too many things going on. I will have to visit our Japanese partners, but a couple days won't hack it. With our opening of a production plant in Ireland, plus developing rep organizations or partnerships in many countries, I'll have to be getting around a lot—at least until we are able to find or develop a general manager to do some of this work.

In case you haven't read, I am very much in need of some help—not only with the publications and books, but in Instant Software. I need technicians, programmers and people with management experience.

If you are interested in further information about going on one of these show tours, watch for more dope in this magazine. Whether I'm along or not won't make a lot of difference, as those on this year's trip discovered. I'm hardly ever available, being rushed from one business meeting to another and giving talks. Sherry and I missed the bulk of the group lunches and dinners on this last trip. We were just kept too busy with meetings and talks.

The most intriguing part of our trip was the few days we spent in Canton, China. Ten tour members went there and had the experience of a lifetime. We all agreed as we returned that we would not have missed it for anything, and that we were very unlikely to go to China again. I'll try to tell you more about it in a later editorial.

JS&A

In the December issue I took JS&A to task for the ROM work on their chess program. I've since received an explanation that they depended upon a Hong Kong supplier, and therein lay much of their problems. Knowing all too well how things work in Hong Kong, I can understand why JS&A might have gotten into some difficulty.

Hong Kong is the land of the rip-off, though the term in their context may have a slightly different meaning than

you may be used to. In this case, it has to do with imitation rather than theft, though the line can be a fine one at times. For example, when Sony came out with their Walkman stereo cassette player, it didn't take the Hong Kong merchants long to come out with a lower-cost copy, much as designer dresses are "ripped off" by the production of bargain copies.

Hong Kong, having very few radio amateurs and therefore a very small supply of engineers and technicians, has to depend upon the brains of other countries for their new products. Their large supply of relatively low-wage help allows them to take the designs and inventions of others and produce them for sale at a lower cost. The drawback is that they are always about a year behind the higher-technology firms. Another drawback is that Hong Kong does not have a governmentally supported program to assure high quality of exports, so you take your chances when you buy a Hong Kong product. There are few known name products from that country. Caveat emptor is the phrase.

Before WWII, Japan was famous for low-cost imitations. Indeed, if something was Japanese, you could be fairly sure that it was of poor quality. The same was true of China at the time. They were noted for cheap, shoddy junk. Today Japan has an image of quality, while products from Hong Kong are suspect.

The lower wages of Hong Kong have encouraged many of the Japanese firms to set up production in the colony. Many American firms, for example, APF, are also doing this. APF is an American firm whose computer is made in Hong Kong.

A visit to Hong Kong is exciting. It is a truly capitalistic country, run primarily by businessmen. Thus, you see few signs of poverty—there are just too many jobs available for anyone to go hungry. The signs indicate wealth. You see more expensive Mercedes cars per block there than anywhere else in the world.

The prices for food are not a lot higher than in the U.S., and they even have a couple McDonald restaurants to satisfy those Big Mac attacks. The city is made up largely of high-rise buildings, which are needed in the limited acreage of the country to house the millions of refugees from China and Viet Nam. The international airport is downtown, since that is about the only flat spot in Hong Kong. The colony is surrounded by China and is dependent upon it for its survival. There is some question as to what may happen in 1999, when the lease on the colony is up. Today China depends a lot on Hong Kong as a bridge to the rest of the world, but if China should really open up, they might not need the colony in ten or 20 years—and the land could return to the control of China. This is an unsettling thought, covered over by brave mutual assurances by the people living there that surely China won't want to upset things. Maybe.

Since business in Hong Kong has been thriving on the brains of others, so to speak, I can easily picture how JS&A got into a problem with their chess game from Hong Kong. Apparently, the chess program involved was on a ROM, and I believe that JS&A had been assured that everything was in order. It turned out that the ROM was copied, and the original owner sued. The first court ruled in JS&A's favor, but on the vague grounds that since the program was not actually in writing, it was not copyright protected. An appeals court again found for JS&A, but on entirely different grounds. The appeals court upheld the copyright protection of programs, whether on ROMs, tapes, disks or whatever. They ruled in favor of JS&A on the basis that the copyright notice was not marked on the ROM. JS&A won, but it was a close one—and you might say it was on a technicality.

Any remarks I've made about the ethics of the situation should be reconsidered in the light of how much JS&A people knew or didn't know about the ownership of the disputed program. I believe Joe Sugarman when he says that he was an innocent bystander in this Hong Kong rip-off. However, I do suspect that he should plead gullible if he trusted the Hong Kong merchants about this. Reason should have told him that Hong Kong has no programmers capable of writing such a program.

Microcomputing Graphic Arts Contest

We know that many of you have been experimenting with computer graphics. So we decided that it is time for you to show us—and your fellow microcomputerists—what you've been up to, with our first *Microcomputing* Graphic Arts Contest. No matter what kind of microcomputer you own, you'll have the opportunity to demonstrate your creativity, imagination and programming skill.

You have four categories to choose from: color video, black and white video, printer hard copy and plotter hard copy. We'll be awarding three prizes in each category: $100 for first prize, a $50 certificate toward the purchase of books from our Book Nook for second prize and a one-year subscription to *Microcomputing* magazine for third prize. We'll also be handing out honorable mentions, and all prize-winning entries will have the chance to appear in the pages of our July issue.

Here are the details:

1. Submit entries in any of the four categories mentioned above: color video, black and white video, printer hard copy and plotter hard copy. Limit: two submissions per category, in as many categories as you like.
2. Submit with *each* entry a card with the following information: Your name, address and phone number; the category of the entry; the title of the entry; a description of the hardware; and the name of any commercial software used in the creation of your work.
3. For the black and white and color video categories, submit glossy prints of the artwork as it appears on your video display. Photos must be at least 5×7 inches.
4. For the printer and plotter categories, submit full-size originals of the work. Entries may be any size.
5. While you don't need to submit programs with your entries, they must be available on request.
6. Naturally, we will accept only those graphics generated on a microcomputer. No minis or mainframes are allowed.
7. Entries must be postmarked no later than April 1, 1981. All judges' decisions will be final. All entries become the property of *Microcomputing* magazine, and cannot be returned. Employees of Wayne Green, Inc., are not eligible.
8. Address entries to Graphic Arts Contest, *Microcomputing* Editorial, Peterborough, NH 03458.
9. All winning entries will be eligible to appear in the July 1981 issue. Who knows: one might even appear on the cover!

One final note. Don't be intimidated if you don't have the latest in fancy equipment. Your creativity and imagination are far more important than the sophistication of your hardware. Remember: a few simple strokes can often express what a thousand cannot.

Articles Wanted

Please keep in mind that I am looking for articles written for businessmen which will tell them what a particular microcomputer has done in a business application. The articles should be written in English and cover everything a businessman wants to know—why the equipment was selected, where the programs came from, how the whole project worked and future plans.

Ditto for school applications for microcomputers. There are thousands of schools which are vitally interested in finding out what is known about the costs and benefits of introducing microcomputers into schools, what can be done with them and what the problems are.

By Robert W. Baker

New PET Monitor

Monjana/1 from Elcomp

Monjana/1

The Commodore monitor provides all the necessary features to examine and modify memory locations or register values, save and load areas of memory, set break points, and so on. However, it lacks several features that could be very useful when debugging machine-language programs.

For instance, there are no disassembler, relocating or tracing facilities, not to mention single-step tracing. Also, there is no real printer support, even though it can be provided by BASIC in a roundabout way. To get printed copies of the output from the Commodore monitor, you first have to issue OPEN 4,4 : CMD 4 BASIC commands in immediate mode before entering the monitor. To turn the printer off, you have to return to BASIC and issue a CLOSE 4 command.

Elcomp Publishing currently offers a well-documented monitor ROM, called Monjana/1, that is an improved version of its earlier Jana monitor for the original 8K PET. The 2K ROM occupies hexadecimal locations $9000 to $97FF using socket UD3 (the right-most unused socket). Unfortunately, this is the same ROM socket used by several other products, but you can always use a ROM switch such as Spacemaker. Monjana/1 user commands provide a number of useful functions for anyone working with machine-language programs:

M = Memory dump. A specified area of memory, between starting and ending addresses, is displayed in a condensed form much like the Commodore monitor. Each display line indicates the hex value of eight consecutive bytes of memory. See Listing 1.

W = Write. Using the W command, you can change or move in memory data edited by an M command.

D = Disassemble. This command is the basis for several more powerful features of the Monjana/1. It displays one machine instruction per line both in hexadecimal and mnemonic code. See the sample in Listing 2.

I = Instruction modification. Using the I command, you can optionally change instructions edited by a D command.

F = Feed. The function of the F command is nearly the same as that of the I command, except that sequential instructions can be entered much easier with automatic sequencing of the cursor and location counter.

T = Trace. Traces program execution, displaying the program counter, registers and stack pointer.

E = Execute. This command executes a single instruction while allowing modification of the program counter, registers or stack pointer. See Listing 3.

J = Jump. This command starts execution of a program at the machine instruction indicated by the current value of the program counter. Execution continues until the first BRK instruction is encountered and control is returned to the monitor. This command is identical to the Commodore G (GO) command.

R = Relocate. Transfers data from one area of memory to another unchanged. The area to be moved is limited by starting and ending addresses, while the starting address is sufficient for the target area. See Listing 4.

```
M:033A-03A2   0  1  2  3  4  5  6  7
W:033A-0341  A5 28 85 1F 85 21 A5 29
W:0342-0349  85 20 85 22 A0 01 B1 1F
W:034A-0351  D0 09 20 95 03 20 95 03
W:0352-0359  4C 39 C4 A2 04 20 95 03
W:035A-0361  CA D0 FA 20 88 03 F0 1A
W:0362-0369  C9 20 F0 F7 C9 8F D0 05
W:036A-0371  20 85 03 D0 FB C9 22 D0
W:0372-0379  09 20 85 03 F0 04 C9 22
W:037A-0381  D0 F7 20 98 03 C9 00 D0
W:0382-0389  DA F0 C1 20 98 03 A0 00
W:038A-0391  B1 1F E6 1F D0 02 E6 20
W:0392-0399  C9 00 60 20 88 03 91 21
W:039A-03A1  E6 21 D0 02 E6 22 60 FF
W:03A2-03A9  00 FF 00 FF 00 FF 00 FF
```

Listing 1. Monjana/1 sample memory dump.

```
D:033A-03A2    1  2  3   MNC-CODE
I:033A        A5 28      LDA $28
I:033C        85 1F      STA $1F
I:033E        85 21      STA $21
I:0340        A5 29      LDA $29
I:0342        85 20      STA $20
I:0344        85 22      STA $22
I:0346        A0 01      LDY =$01
I:0348        B1 1F      LDA ($1F),
I:034A        D0 09      BNE $0355
I:034C        20 95 03   JSR $0395
I:034F        20 95 03   JSR $0395
I:0352        4C 39 C4   JMP $C439
I:0355        A2 04      LDX =$04
I:0357        20 95 03   JSR $0395
I:035A        CA         DEX
I:035B        D0 FA      BNE $0357
I:035D        20 88 03   JSR $0388
I:0360        F0 1A      BEQ $037C
I:0362        C9 20      CMP =$20
I:0364        F0 F7      BEQ $035D
I:0366        C9 8F      CMP =$8F
I:0368        D0 05      BNE $036F
I:036A        20 85 03   JSR $0385
I:036D        D0 FB      BNE $036A
I:036F        C9 22      CMP =$22
I:0371        D0 09      BNE $037C
I:0373        20 85 03   JSR $0385
I:0376        F0 04      BEQ $037C
I:0378        C9 22      CMP =$22
I:037A        D0 F7      BNE $0373
I:037C        20 98 03   JSR $0398
I:037F        C9 00      CMP =$00
I:0381        D0 DA      BNE $035D
```

Listing 2. Monjana/1 sample disassembly of area dumped.

```
T:<PC> SR AC XR YR SP   MNC-CODE
E:033A 30 00 30 01 02   LDA $00
E:033C 30 4C 30 01 02   STA $1F
E:033E 30 4C 30 01 02   STA $21
E:0340 30 4C 30 01 02   LDA $29
E:0342 32 00 30 01 02   STA $20
E:0344 32 00 30 01 02   STA $22
E:0346 32 00 30 01 02   LDY =$01
E:0348 30 00 30 01 02   LDA ($1F),Y
E:034A 32 00 30 01 02   BNE $0355
E:034C 32 00 30 01 02   JSR $0395
E:0395 32 00 30 01 00   JSR $0388
E:0388 32 00 30 01 FE   LDY =$00
E:038A 32 00 30 00 FE   LDA ($1F),Y
E:038C 30 07 30 00 FE   INC $1F
E:038E 30 07 30 00 FE   BNE $0392
E:0392 30 07 30 00 FE   CMP =$00
E:0394 31 07 30 00 FE   RTS
E:0398 31 07 30 00 00   STA ($21),Y
```

Listing 3. Monjana/1 sample trace of program execution.

PERMANENT RELIEF

Of today's and tomorrow's Word Processing problems

Apple PIE + Formatter

Apple PIE (Programma International Editor) nd FORMAT (text formatter) offer full strength olutions to today's word processing problems. 'hese versatile, powerful programs provide locument preparation and word processing apabilities previously found only on much larger omputer systems.

PIE is a general purpose, full screen editor nat uses control keys and function buttons to rovide a full range of editing capabilities such as earch and replace, delete, copy, insert, move. hanges may be made directly anywhere on the creen and are shown as they are performed.

FORMAT uses simple instructions nbedded in the input text to describe the desired ppearance of the final document. It handles entering, underlining, indenting, page numbering, margins, headers, footers, even form letters, and includes a proofing capability.

These high-quality, cost-effective programs come with comprehensive documentation and run on a 32K Apple II. They are available through your local computer store or direct from Programma International, Inc. at the introductory price of $79.95*.

VIDEX VERSION T.M.
DOUBLE VISION T.M.
SUPR TERM VERSION T.M.
STANDARD VERSION

*December 1, $129.95.

PROGRAMMA ✓11

3400 Wilshire Boulevard
Los Angeles, California 90010

or

Simple enough for the beginner. Versatile enough for the professional.

```
A:033A-03A2 TO 0800-0868
R:033A-03A2 TO 0800-0868
N:
N:RELOCATE PROGRAM
N:
M:0800-0868    0  1  2  3  4  5  6  7
W:0800-0807   A5 00 85 1F 85 21 A5 29
W:0808-080F   85 20 85 22 A0 01 B1 1F
W:0810-0817   D0 09 20 95 03 20 95 03
W:0818-081F   4C 39 C4 A2 04 20 95 03
W:0820-0827   CA D0 FA 20 88 03 F0 1A
W:0828-082F   C9 20 F0 F7 C9 8F D0 05
W:0830-0837   20 85 03 D0 FB C9 22 D0
W:0838-083F   09 20 85 03 F0 04 C9 22
W:0840-0847   D0 F7 20 98 03 C9 00 D0
W:0848-084F   DA F0 C1 20 98 03 A0 00
W:0850-0857   B1 1F E6 1F D0 02 E6 20
W:0858-085F   C9 00 60 20 88 03 91 21
W:0860-0867   E6 21 D0 02 E6 22 60 FF
W:0868-086F   11 FF A9 0E 20 AE F2 68
N:
N:NOW RELINK PROGRAM AT NEW LOCATION
```

Listing 4.

L and A = Link and Addressing. The L command adjusts all the relative addresses in a machine program as necessary after being relocated. The monitor searches the link area for valid instructions and tests if these instructions contain addresses which relate to the transferred memory area. If so, the monitor adjusts the address to correspond to the relocation. Before executing the L command, the monitor shows the addresses used with the last R or A command. By use of the A command, these addresses may be changed on request. See Listing 5.

One convenient feature of the L and A commands is the ability to create a "where used" list for selected addresses or subroutines. Whenever the starting address is 0000, no data is actually changed by the L command, but the information lines are still displayed.

S = Save. Saves an area of memory to tape with a filename of up to 14 characters. You cannot, however, save memory to disk as you can with Commodore's monitor. Also, there is no load command with Monjana; you must load via BASIC.

P = Print. Enables or disables echoing a copy of all monitor output to a printer or any IEEE device between addresses 4 and 7. This is a convenient feature, especially with the disassembly capabilities.

N = Note. Allows adding comment lines to the printer output wherever desired.

X = Change to BASIC. Exits the monitor and returns control to the BASIC operating system.

In addition to the relocating and disassembly features, Monjana also "pages" all screen displays for greater convenience. Whenever a full screen of information is displayed, a continue line is displayed. The monitor then waits for you to press the return key before continuing the current function.

All in all, the monitor is convenient and easy to use. It appears to be very well written, includes a number of user-oriented features and is of great value to any serious machine-language programmer. Many other extended monitors exist, but this is the only one I know of available in ROM. I'm not sure, however, if it will function properly with BASIC 4.0; I suspect that it will not work with the new ROMs.

```
L:0800-0868
A:033A-03A2 TO 0800-0868
I:0812          20 5B 08    JSR $085B
I:0815          20 5B 08    JSR $085B
I:081D          20 5B 08    JSR $085B
I:0823          20 4E 08    JSR $084E
I:0830          20 4B 08    JSR $084B
I:0839          20 4B 08    JSR $084B
I:0842          20 5E 08    JSR $085E
I:084B          20 5E 08    JSR $085E
I:085B          20 4E 08    JSR $084E
N:
M:0800-0868    0  1  2  3  4  5  6  7
W:0800-0807   A5 00 85 1F 85 21 A5 29
W:0808-080F   85 20 85 22 A0 01 B1 1F
W:0810-0817   D0 09 20 5B 08 20 5B 08
W:0818-081F   4C 39 C4 A2 04 20 5B 08
W:0820-0827   CA D0 FA 20 4E 08 F0 1A
W:0828-082F   C9 20 F0 F7 C9 8F D0 05
W:0830-0837   20 4B 08 D0 FB C9 22 D0
W:0838-083F   09 20 4B 08 F0 04 C9 22
W:0840-0847   D0 F7 20 5E 08 C9 00 D0
W:0848-084F   DA F0 C1 20 5E 08 A0 00
W:0850-0857   B1 1F E6 1F D0 02 E6 20
W:0858-085F   C9 00 60 20 4E 08 91 21
W:0860-0867   E6 21 D0 02 E6 22 60 FF
W:0868-086F   11 FF A9 0E 20 AE F2 68
```

Listing 5.

Monjana/1 for the 2001 series PET/CBMs is $98, complete with documentation. You can order the manual alone for $3.95. For more information, contact Elcomp Publishing, 3873L Schaefer Ave., Chino, CA 91710.

Elcomp Publishing also offers a book titled "Care and Feeding of the Commodore PET" for $11. Eight chapters cover various aspects of the PET hardware, complete with schematics and IC specification sheets. Although written primarily for the original 8K PET, it may be of limited interest to new PET owners.

BASIC 4.0/DOS 2.1

Many people have recently written with questions about the new BASIC 4.0 and DOS 2.1 ROM upgrades. It appears that they are confused about the changes and what's actually involved in upgrading. Let's examine the upgrades and their related changes.

Anyone with a 2040 disk can upgrade to DOS 2.1 ROMs without upgrading to BASIC 4.0. The original DOS Wedge program will still work with the new disk ROMs. If you upgrade your PET/CBM to BASIC 4.0, then you *must* upgrade your 2040 disk to DOS 2.1 for the system to operate correctly.

When you've upgraded to either BASIC 4.0 or DOS 2.1, you should not have to make any major changes to BASIC programs that use sequential disk files. You do not have to use the new disk commands provided by BASIC 4.0; all the original disk commands should still operate correctly. If any programs use random access files, then you may have a job on your hands. The handling of random access files has been changed quite a bit.

Otherwise, the only problem area I've seen is that the Block Allocate (BA) command does not return the track and sector of the next available block when attempting to allocate a block that is not available. Also, several features of the DUM 3.4 program supplied with the 2040 disk no longer work with the new disk ROMs.

Be aware that any disk formatted by DOS 1.0 cannot be duplicated, verified or written to once the DOS 2.1 ROMs are installed. This means you cannot create backup copies of any DOS 1.0 formatted diskettes once you change ROMs. Because of this, I strongly suggest copying all your diskettes onto newly formatted diskettes (formatted using DOS 2.1) and keeping the DOS 1.0 disks until you are absolutely sure the new disks are OK. You may even want to keep the original DOS 1.0 disks as master backups and file them away.

A new disk command has been included in DOS 2.1 to aid in copying your disks. All you have to do is format the new disk and then issue a copy command without any filenames: COPY D0 TO D1 using BASIC 4.0 or C:D1 = D0 with BASIC 3.0. This will copy every file from the old disk to the new disk. Be patient, this process takes much more time than a disk duplicate command, but it sure beats typing in a copy command for every file.

You can even take this opportunity to reexamine your assignment of disk IDs. It is now very important to use a unique ID for every disk. DOS 2.1 uses the disk ID to determine when a disk has been changed in a drive, so it knows when to initialize a drive. You no longer have to initialize a drive when you insert a disk; it's done automatically for you by DOS.

However, if two diskettes have the same ID, DOS will not know when to reinitialize the drive. Thus, it will continue to use the original block availability map (BAM) for the second disk. This will totally destroy the second disk when the wrong BAM is written to it.

Normally, this will not present a problem if you take care when assigning new disk IDs. However, when a backup copy of any disk is created, you now have two (or more) disks with the same ID. Be careful in what order you insert diskettes or issue the old initialize command.

If necessary, the auto-initialization function can be switched off by issuing the following commands:

```
OPEN 1,8,15
PRINT#1,"M-W";
CHR$(243);CHR$(16);CHR$(1);CHR$(1)
```

Since you are recopying all your disks anyway, you may want to reorganize a little and put different files together on the same disk.

You may have another problem in this area if your old format disks are almost full. Newly formatted disks created with

DOS 2.1 will have six less free blocks than disks formatted by DOS 1.0. Thus, all your files from the old disk might not fit on the same new disk. Also, the total number of files has been reduced to 144 files maximum on any given disk.

I haven't had time yet to track down any low memory changes in the new BASIC, but I haven't heard of any changes in that area. However, the ROM subroutine addresses have been changed. Some of the changes are for the better, since a number of new subroutine pointers have been added at the top of memory. Just keep in mind that the ROMs have changed, and any PEEK, POKE or machine-language routines/programs are likely not to work. The new disk commands in BASIC 4.0 makes use of certain areas in the second cassette buffer for work areas. So be careful when you use utility programs written for the second cassette buffer; they may be destroyed by disk operations.

Another area of change is the addition of two new reserved variables—DS and DS$ for disk status. You can no longer assign values to these variables since they are now reserved variables assigned values by the system. Any program that used these variable names for its own purpose must be changed to use new variable names.

The variables DS and DS$ now contain the values normally read via the disk error and control channel, channel 15. They are automatically updated whenever referenced, after a disk operation is performed. Thus, you no longer have to read the error channel to see if an operation failed. Simply check the error number in DS or the message in DS$.

While on the subject of new ROMs, let's take a quick look at the new BASIC commands. In general, parameters for the new commands are not order dependent; the device, filename and unit will work in every permutation. Filenames may be in quotes or represented as a string variable.

Drive numbers are indicated by the letter D, followed by 0 or 1. The default drive number is generally 0. If you can specify more than one filename in a command, the default drive number for the second drive will be the last user-specified drive number. The unit number is optional on nearly all commands and defaults to 8. It is specified by the letter U, followed by an integer between 4 and 31 to identify the IEEE device number. Variables may appear as parameters, but they must be enclosed by parentheses, for example, U(A+2).

DOPEN #lfn,"filename" [,Ly] [,Dx] [ON Uz] [,W]—opens a disk file with the specified logical file number (lfn). Optionally, the file can be a random access file if a record length (y) is indicated. The file is normally opened for read operations unless the W option is specified.

DCLOSE [#lfn] [ON Ux]—closes a specified logical file or all disk files if a logical file number (lfn) is not given.

RECORD #lfn,r [,b]—is executed directly before any GET#, INPUT# or PRINT# to position the disk at the desired record (r) of a random access file. If this command is not executed, then the position accessed will be the next record directly after the last referenced record. Optionally, the exact character position (b) within the record can also be specified. This new command is transmitted to the disk control channel (channel 15) as a five-byte message. This message consists of the letter P, followed by the logical file number, the record number as a two-byte value and the character position.

HEADER "name",Dx [,Izz] [ON Uy]—performs the former NEW command to format a diskette with the name and ID (zz) specified. The advantage of using the new BASIC command is that the command requests a confirmation that you really want to format the disk. You must respond with "YES <RETURN>" for the command to execute.

COLLECT [DX] [ON Uy]—issues the former VALIDATE command to free up allocated space of improperly closed files on the disk. It also deletes their references from the directory and verifies the BAM stored on the disk.

BACKUP Dx TO Dy [ON Uz]—duplicates disk x onto disk y complete with disk name, ID, file layout and contents. This is the former DUPLICATE command, unchanged but much faster than before.

COPY [Dx] "Name-1" TO [Dy,] "Name-2" [ON Uz]
COPY Dx to Dy [ON Uz]—copies disk files just as before. One new option is the ability to copy all files from one disk to another with one command.

CONCAT [Dx,] "Name-1" TO [Dy,] "Name-2" [ON Uz]—concatenates one file (Name-1) to the end of a second file (Name 2). It issues the former COPY command with the appropriate arguments.

DSAVE "Name" [,Dx] [ON Uy]—saves a BASIC program on disk just as before with the SAVE command.

DLOAD "Name" [,Dx] [ON Uy]—loads a BASIC program from disk just as before with the LOAD command.

DIRECTORY [Dx] [ON Uy]—displays the disk directory from the specified drive (x).

RENAME [Dx,] "Name-1" TO "Name-2" [ON Uy]—changes the name of a disk file just as with the old command.

SCRATCH [Dx,] "name" [ON Uy]—deletes a disk file just as with the old command. However, with BASIC 4.0 a confirmation must be made before the file is actually scratched. This confirmation is only needed if the SCRATCH command is used in the direct mode.

APPEND #lfn, "name" [,Dx] [ON Uy]—allows you to open a previously written sequential file with the write pointers positioned to the next character position beyond the end of file. This allows adding data at the end of an existing data file without rewriting the entire file.

One last change in BASIC is the redefinition of the SHIFT-RUN sequence. This combination of keys now loads the first file from the disk in drive 0 instead of loading from tape 1.

I hope this list of commands has been helpful. The ROM upgrades seem to be well worth the effort: the new disk format is much more reliable than before; disk duplicating (for backups, etc.) is now much faster; the "garbage collection" is hardly even noticeable; and the new disk commands are more convenient than using the Wedge. Current 2040 disks are being shipped with DOS 2.1 installed, and a new 4000 series of 40-column PETs are being shipped with BASIC 4.0 installed.

Odds and Ends

Benson Greene (210 Fifth Avenue, New York, NY 10010) has an extensive accounting program available for only $35 on tape. This program was written for an 8K PET with new ROMs, 32K on Expandapet memory, 2040 disk and an Axiom printer. Conversion for other equipment should not be difficult, according to Greene. The program, called MASTER1, produces accounts payable/receivable, purchase order, acknowledgement, invoice, packing slip, voucher, statement, checkbook, cashbook, price lists and account status.

In addition, the program will design and produce any other type document required. Other features are proposed for a newer version designed for the 8032 CBM. The author warns the program is slow, but it still may be of interest.

The Central Illinois PET Users (c/o Jim Strasma, 3838 Benton Drive, Decatur, IL 62526) are offering free copies of their new *Midnite Software Gazette*. Simply send a self-addressed envelope with two stamps for return postage. Thereafter, simply send a replacement envelope with postage whenever you receive another issue. Their newsletter will contain brief reviews of various new programs and hardware for the PET/CBM.

I stand corrected on a comment in my October column! A reader in Texas wrote that the PET was actually seen on the "Super Train" TV show before appearing on "Pink Lady." Have you seen the PET anywhere else?

Correspondence concerning this column should be addressed to Robert W. Baker, 15 Windsor Drive, Atco, NJ 08004.

By Walter Koetke

Educators Are Buying

Software publishers Eye schools As a marketplace

But What about the Software?

The potential of the educational establishment as a marketplace for microcomputer software has attracted the attention of many corporations and a host of individual software developers. Most of those attracted are very interested in predicting the future of this market. While their predictions vary, there is general consensus that the size of the marketplace is already significant, with an expected growth rate that is quite impressive.

One generally conservative source places 1980 school expenditures for computer hardware and software in excess of 90 million dollars. Roughly 30 percent, or 27 million dollars, represents the software portion of this expense. The same source projects that educators will spend nearly 800 million dollars for computer hardware and software in 1985. Approximately 60 percent of that amount, or 480 million dollars, will be spent on software.

These figures illustrate that the annual educational expense for computer hardware and software in 1985 will be more than eight times this year's expenses. And of perhaps even more significance, the annual expense for software in 1985 will be almost 18 times that of this year. But what about the software? On what software should the money be spent?

Selecting the type of computer hardware most appropriate for educators today is not an easy decision. This topic was discussed in last month's column. It is interesting to speculate what hardware configurations will be available in 1985. The industry is changing rapidly. The two microcomputers most prevalent in schools today are the Model I TRS-80 and the small keyboard PET. Both of these machines are already out of production.

There does, however, seem to be one rather safe prediction—there will be lots of microcomputers in a variety of configurations. Seymour Papert's current work that provides individual microcomputers for each member of a third grade class will no longer appear to be a unique, idealized experiment. I suggest that students at most grade levels will have easy access to computing facilities. The existence of computer laboratories that can provide one computer for each member of a class is no longer unusual at the secondary school level and rather common at the college level. Spokesmen for Pittsburgh's Carnegie-Mellon University have vowed that by 1985 every freshmen enrolled in the university will be issued a microcomputer as part of the university's orientation package. Certainly many other institutions will follow the lead of Carnegie-Mellon.

Dr. Papert has observed that purchasing microcomputers to be issued to each first grader would add very little to the present expense of educating a child for 12 years. When you consider the educational potential of this expense, you might easily assert that the initial first grade cost increase would result in actual savings over 12 years. Microcomputers will be in the hands of perhaps the majority of college students, will be readily available to most secondary students and will be accessible by many elementary students. But what about the software?

The National Council of Teachers of Mathematics' highly regarded "Recommendations for School Mathematics of the 1980s" offers eight major agenda for action. The third recommendation is that "mathematics programs must take full advantage of calculators and computers at all grade levels." The NCTM goes on to explain that computers should be used in imaginative ways for exploring, discovering and developing mathematical concepts and not merely for checking computational values or for drill and practice. Similar recommendations can be expected in subject areas as well. But what about the software?

Let's take a look at the educational applications of today's microcomputers. These are all available right now. While much of their potential has yet to be utilized, all can be used in the classroom tomorrow or next week.

1. *Computer literacy*—one of the most urgent applications. Quite simply, computer literacy means learning what computers can and cannot do, learning how to write simple programs and learning that blaming the computer is rarely a valid excuse. Dr. Molnar of the National Science Foundation made a statement that might be paraphrased as "Any teacher who still thinks he or she might be replaced by a computer, should be." Dr. Molnar quite correctly feels that computer literacy must be expected of all teachers of any subject at any level. But computer literacy carries some changes in the educational arena. A computer literate secondary student will not accept the standard answer each September that his high school schedule was messed up by the computer.

2. *Teaching programming*—in my opinion, appropriate for all grades beginning with third. Seymour Papert recently demonstrated that nonreading preschoolers can learn to program, but his results are well ahead of the average classroom. Unless you're teaching programming as a part of a computer science course, I suggest you use BASIC, not because BASIC is the best of all languages (it certainly is not), but because the wealth of supporting material is readily and inexpensively available. This is not the case for any of the other languages often proposed as alternatives.

3. *Computer science*—the study of the computer and related ideas for their own sake. Students learn about the computer

Walter Koetke, Putnam/North Westchester BOCES, Yorktown Heights, NY 10598.

itself rather than using the computer in a variety of other areas. While the future may evolve a different set of priorities, I suggest that today's computer science is most appropriate and effective at the high school or college level.

4. *Providing a tool for the gifted*—an absolute necessity for any school system that is properly serving this special population. These are the students on whom our society will depend in the future, and we will all benefit if they are provided early access to the intellect-expanding capabilities of the computer. Stories of bright students and their computer feats are already legendary. The largest primes and the largest perfect numbers have all been discovered by bright students with access to computers. Repeatedly, these students demonstrate their inability to recognize a truly difficult or seemingly impossible problem by solving the problem. For the sake of all of us, let's see that they have full access to whatever tools they require.

5. *Supporting special-education students*—an application that produces positive results in the vast majority of cases. Its publicized potential often refers to technical types building special devices, but personal observations of teachers and special-education students using standard microcomputers and peripherals with teacher-generated programs suggest an even more impressive potential. There are many theories as to why so many special-education students respond positively to computer use, and those may be the subject of a future column. Providing a microcomputer for an interested teacher of special-education students is, however, a productive action regardless of the theories.

6. *Drill and practice*—often mundane, sometimes boring, and certainly an unaesthetic use of a powerful tool, but it works! The microcomputer can add several important dimensions to drill and practice. Most obvious are the immediate interaction and reinforcement provided to the student and the record-keeping aspects available to the teacher. Less evident but far more important is the ability to tailor drill and practice to meet the needs of individual students. Check last November's column for some examples of drill and practice personalization.

7. *Problem solving*—using the computer as a tool in one or more of the steps required to solve a problem. This was done effectively in mathematics and science classes for many years using time-shared terminals. With the additional capabilities available with today's microcomputers, problem solving should also become a viable application in several additional subject areas.

8. *Simulations*—using the computer to represent those things that are too hard, too time consuming, too expensive or too dangerous to do in reality. Simulations help answer "what if?" questions in a safe, timely and cost-effective manner. Effective uses of simulations were, for example, responsible for minimizing the danger as Skylab fell and as Three Mile Island overreacted. The design and use of simulations are becoming increasingly important in many fields, and understanding their proper applications, strengths and weaknesses is a skill we should impart to all students.

9. *Games*—have positive aspects from motivational carrot, to exemplary programming techniques, to developing problem-solving skills, to disguising learning environments in a variety of enjoyable forms. There are not many who still resist the value of appropriate use of games in a learning environment.

10. *Text editing*—not really something that can be done tomorrow in most schools. The hardware and software exist, but there remain two obstacles. Text editing assumes the availability of hard copy, but printers are not yet widely available in schools. Of even more importance is the fact that text-editing capabilities and techniques are foreign to most staff members, and their skills must be updated before they can pass those skills to their students. I believe that text editing will result in dramatic changes in the effectiveness of teaching English and writing skills. Students in third and fourth grade can effectively use text editors, and providing text-editing capabilities to students throughout their school experience should soon become an important goal.

The preceding list of applications is hardly controversial. Given the ability to add or delete just one application, I believe most would find the list complete and accurate. The list really does represent those applications that teachers, depending upon their own experience, are either already using or attempting to use. But what about the software?

Let's examine the world of microcomputer software as it pertains to education. There are certainly many developers of educational software. What are they doing?

At one end of the spectrum is Seymour Papert using the LOGO language with young children. LOGO is the only major language that was written specifically for children. Other more available languages such as BASIC are being used not because they are best for children or education, but because they were in the right place at the right time. Educators chose the best of what was already available rather than creating a new language that truly met the needs of the children. This utilization choice is not inherently bad. In fact, it was the best choice at the time. The only danger is that the educational community may become so absorbed in currently accessible languages that they become reluctant to adopt the major advantages of new languages becoming available.

The thrust of the work with LOGO is students writing their own programs—item 2 of our previous list. Their programming ability is then used as a major tool in teaching mathematics, logic, problem solving and even reading. Thus, the leading edge of educational software development lies in the area of language development appropriate for allowing very young students to write their own programs. A future column will be devoted to a detailed discussion of LOGO.

At the other end of the spectrum are the names of publishers long familiar in classrooms across the country. Their interest in educational software is quite evident—corporate profits. This statement is offered as fact, not criticism. This group is doing all software development on plug-in ROMs.

Their ad men are doing an excellent job. "By giving you ROMs we're protecting you. The software is unalterable and can't be messed up by the student." Sounds good until you think about it. The ability to modify software to meet the specific needs of individuals is an important characteristic of computers in education. you can modify the data, change the reading level, alter the rewards, personalize, and on and on, but not with unalterable software.

"We want you to be comfortable. If our program doesn't work, then the program is broken and you can return it for a replacement." Once again, it sounds good until you think about it. A program that cannot be improved does not exist. Programming is not a right or wrong skill. Solutions are evolved. Solutions can be improved. Giving the impression that the computer programs are either working or broken does not even contribute to computer literacy.

I suggest that just as the traditional approaches of traditional Wall Street firms are being challenged and often beaten by a new generation of individuals with reasonable knowledge and access to computers, so will the publishers be severely challenged. The resulting shake-up should be fun to watch. More importantly, I believe the students will be the winners. Hence the trailing edge of educational software development now lies in the offices of traditional educational publishers.

What should educators do tomorrow? I suggest that you are well advised to learn to program. Certainly all teachers will not develop large, original programs. However, I believe the majority of teachers can and should learn to program well enough to either modify existing software themselves or to direct others who modify existing software so it is tailored for specific classes or individuals. I also suggest that teachers acquire a software library of material appropriate for classroom use, but only if they're able to obtain both the program code and the ability to modify that code.

By Frank J. Derfler, Jr.

Coming In 1981

Talking terminals, software by phone on the horizon

The microcomputer world develops a different new product emphasis every year. In 1978 we found many different kinds of solid-state memory devices and boards, 1979 brought us a host of different terminals, and 1980 has to be remembered as the year of the hard disk.

Talking Terminals

What will we see evolve in 1981? My best guess is talking terminals. I know we have had voice synthesis and voice command recognition for quite a while, but the systems have been limited in capability and quality. This time, the consumer appliance field may be leading us into a new development that will (again) change the way we do things—even our data communications. Talking pinball machines, microwave ovens and television sets are about to enter on the scene. Can a talking terminal be far behind?

I am not predicting that computer terminals in the early part of the 80s will have the speech I/O capability of Captain Kirk's computer on the Starship Enterprise. Keyboards will still be a primary means of data entry, and we need to teach our kids keyboard skills. But speech synthesis and recognition will augment keyboards and be used for certain special-purpose terminals.

The average terminal could recognize certain voice commands, just as it now recognizes specialized keys and control codes. Terminals may have voice-recognition "extensions" that allow you to input data via microphone when you are away from the keyboard. This feature would be useful for inventory control, production lines, military command posts and other busy areas.

Lear Siegler, Inc., has recently announced plans for a speech-recognition system for its popular ADM-3A terminal. Terminals and peripherals may talk to you to get your attention. Printers may gently remind you of "low paper," and your terminal may tell you the computer is "ready" or "finished."

Speech I/O can come into its own in the areas of games and education. How about a game of Adventure where lions snarl, swords clash and monsters howl? How about a math drill program with spoken words of praise, applause and gentle correction? These things and many more are possible.

There are three competing methods of performing speech synthesis. The Votrax division of Federal Screw Works uses a principle called format synthesis, which relies on stored bits of words called phonemes and allophones. An early version of this kind of system is available for the TRS-80.

Texas Instruments uses computer analysis of how words are formed in the human vocal system to reproduce those words under a system called Linear Predictive Coding. Most of us have probably heard their Speak and Spell device.

National Semiconductor Corporation uses sophisticated waveform digitization, which is compressed to save memory space.

I won't go into any of the technical details, but if you are interested, the May 22, 1980, issue of *Electronics* carried more in-depth information about the systems. The main point is that these systems can store and reproduce 100 or more words or sounds using a fairly typical microcomputer system.

Speech recognition is a more difficult job, but it can be done well and economically with a vocabulary of about 100 words. Storage for 100-word reference patterns might use between 8 and 16K of RAM. You can do a lot of commanding and inputting with 100 words.

Watch for speech I/O devices. I am sure they will find uses we can't see right now, but they will certainly add to our communications capabilities.

Software by Phone

Have you noticed the trend toward shopping by telephone? I like it, especially when retailers supply an 800 toll-free number. I have done business with many of *Microcomputing*'s advertisers by phone, and the response and personal attention are rewarding. It had to come to pass that software could not only be ordered but also delivered over the phone.

If you have an Apple computer, you can buy software by phone and have instant delivery from the Telephone Software Connection (213-329-3715). You need an Apple with disk, Applesoft and the Hayes modem card. When you first dial into this system, you establish an account with credit-card billing reference. It takes one business day before your account is verified. After this initial verification, you select a password to let you buy software at any time.

Telephone Software Connection has some unique programs, most of which are aimed at data communications. Their prices range from $20 for some two-player games (one player can be remote) to $45 for a data transfer program that can transfer binary files from one Apple to another. Most programs can be completely purchased during a $2 weekend phone call.

The Telephone Software Connection has competition from free software exchanges, but their unique programs give them an edge. We are learning to beat the price of gas and the frustration of mail by squeezing more and more down our phone lines; this is part of the fun and value of dialing-up.

Spotlight

Amateur Radio Research and Development Corporation (AMRAD) is a nonprofit corporation in the Washington, D. C., area made up of folks interested in both microcomputers and amateur radio. This active and innovative group runs both an amateur radio two-meter repeater and a computer message system. They also publish the most interesting newsletter on computer and amateur communications I have found.

Address correspondence to: Frank J. Derfler, Jr., PO Box 691, Herndon, VA 22070.

Two Great New TRS-80 Computers. Radio Shack—Setting New Standards For Price and Performance!

13-Inch Color TV Receiver Available Separately. Only $399*

16K TRS-80 Model III. The new standard in personal computers—packed with the advanced features you want, yet priced unbelievably low! The Model III's handsome *one-piece* cabinet includes a high resolution 12-inch monitor that displays 16 lines of 64 upper/lower case characters, and graphics. It also includes a 65-key typewriter-style keyboard with data pad. There is room inside for two optional double-density disk drives—and you can add two more external drives for a total of approximately 670K bytes of on-line storage. Model III comes with a two-speed cassette interface, so most of our huge library of Model I, Level II software can be used. And the new 1500-baud mode gives you really fast cassette I/O. The very powerful Model III BASIC language, contained in 14K of ROM, features automatic line numbering, tracing, PEEK and POKE, machine language programming, keyboard-controlled screen print, I/O device routing, repeating keys, special characters and lots more. Other Model III features include a built-in real time clock, cassette jack and Model III bus, plus a built-in printer interface for direct connection with Radio Shack printers. With the built-in RS-232 firmware and the optional internal RS-232 board, communications is a snap! Model III is expandable to 48K RAM at any time, and is also available with 4K RAM and Level I BASIC (for only $699*). A full-blown 32K, 2 disk system with RS-232 is yours for only $2495.* This is the breakthrough you've been waiting for. Come in and try Model III today. **16K TRS-80 Model III . . . . Only $999***

16K TRS-80™ Color Computer. Vivid 8-color graphics, sound, the high performance 6809E microprocessor and more—at a very affordable low price! This amazingly portable new micro attaches in seconds to any TV set. Use it at work, at school or at home. Its Extended Color BASIC language provides unique commands for graphics manipulations, easy machine language programming, complex musical sounds (with more than one note!), PEEK and POKE, specific error messages and much more. A built-in RS-232 serial interface lets you attach a modem for communications or a serial printer. Of course, you can save your programs on cassette (at a fast 1500 baud)—or just pop in one of our instant-loading ROM Program-Paks for a wide variety of fast-paced games and valuable personal applications. Like all TRS-80s, it's easy to expand—add a printer now, disk drives are on the way. An optional pair of joysticks brings extra maneuverability and real excitement to computer games. The TRS-80 Color Computer is also available with 4K RAM and Color BASIC (at just $399*). This is the multipurpose computer you've been waiting for. Try it today at your nearest Radio Shack store, Authorized Dealer or Computer Center. **16K TRS-80 Color Computer with Extended Color BASIC .Only $599***

Radio Shack®

The biggest name in little computers™

A DIVISION OF TANDY CORPORATION

*Retail prices may vary at individual stores and dealers. Ask for our FREE 1981 TRS-80 Computer Catalog or Write Dept. 84, 1300 One Tandy Center, Fort Worth, Texas 76102

It had to come to pass
that software could not only be ordered
but also delivered over the phone.

If your interests fall across these two areas, you should contact AMRAD at 1524 Springfield Ave., McLean, VA 22101. Their newsletter is $12 a year.

The AMRAD computer message system was probably the fourth one in the nation. The original system is now off the air. It has been replaced by a very high-quality CBBS run by Terry Fox (703-734-1387), but I would like to briefly describe the old system because of its unique design.

The AMRAD system serves as a model of what can be done with very little. The hardware consisted of an AMI computer board using a 6800 CPU, a keyboard and monitor, 16K of RAM and a much-converted answer-only modem. That's all—no whirring disk drives and great stacks of program options.

The purpose of the system was to beat what I have named the "Time Tyranny of Telecommunications." The little system left and received messages at convenient times. The message program was written in machine language and occupied a tiny space in RAM. A small HELP listing was available, but otherwise the system only asked for your command. The message numbers were even in hex to save program space. All messages resided with the program in RAM.

This method of providing a message service had both good and bad points. It was very sensitive to lightning and power failures and had no back-up; if it crashed, the slate was wiped clean. RAM space was limited, so everyone had to leave short notes and clean their messages off frequently.

This required a little club discipline, but it was easier to do than you might think because the brevity of the commands and instructions promoted brevity in the messages. It was inexpensive and simple. With this design philosophy, you could certainly get a system operational using a KIM or similar single board computer for a very minimum expense.

The message system was also tied into the amateur radio repeater system. Local hams could access the system using Murray (Baudot) code in the two-meter amateur band. (Murray code was used because transmission of ASCII was not legal then.) This access proved to be redundant to the phone line because of the local nature of the radio link, and it was eventually removed to save program space. This was an innovative idea and might see more use now that ASCII transmission is legal on the high-frequency bands.

AMRAD also runs a separate but identical system for the deaf in Virginia called the Virginia TTY. Users with Murray terminals can reach this service at 703-765-2161.

Another important AMRAD project is the Handicapped Educational Exchange (HEX). This is a bulletin board system set up by AMRAD and supported by a federal grant from the Bureau of Education for the Handicapped. It will accept inputs from either a 60 wpm Murray terminal or 110/300 baud ASCII terminal.

HEX runs on a Smoke Signal Broadcasting computer with a 6800 CPU. It uses two eight-inch double-sided floppy disks for storage. The program is similar to the CBBS format, but with explanations worded for the nontechnical users. It is written in BASIC and runs in 32K. The system can be reached at 301-593-7033.

Because of its unique character, the messages left on this system should relate to the handicapped, education or communications and microcomputer technology.

AMRAD has also done a lot of work in interfacing the Murray TTY terminals used by the deaf with microcomputers. Two names stand out: Bob Bruninga and Paul Rinaldo. Bob is gone from the area, but his work and contribution cannot be forgotten. Paul can be reached through AMRAD.

Datacomm Fun

Many different forms of data communications are available to you as a private individual. They range from local message systems, to commercial systems such as The Source, to sending your messages out on high-frequency amateur radio bands. If you have any interesting notes or experiences in these areas, drop me a line. Send paper mail to PO Box 691, Herndon, VA 22070, and include a stamped envelope if you expect a reply. Send electronic mail to TCB967 on The Source, 70003, 455 on CompuServe or the AMRAD CBBS.■

By Eric Maloney

Computer Talk

A new column takes a look at computers and language.

In 1789, Benjamin Franklin penned a letter to Noah Webster praising the "Zeal for preserving the Purity of our Language" that Webster exhibited in his predictionary work, *Dissertations on the English Language.*

But Franklin, notwithstanding his predisposition for innovation, took Webster to task for approving several new words that he felt deserved severe reprobation. In particular, he called onto the carpet four abominable verbs: notice, advocate, *oppose and progress.*

Of course, if an editor today took exception to these words, his boss would probably suggest a long vacation. But change is one of the outstanding characteristics of language. What was unacceptable yesterday is used without a second thought today, and what is common today will be an archaism tomorrow.

This is particularly true of the American language. From the beginning, we have displayed an extraordinary drive to mangle, reshape and invent the tools with which we express ourselves.

This lust for linguistic mutilation grew largely from our compulsion to distinguish ourselves as a distinct cultural organism. Our ancestors deliberately flaunted their bastardizations of the good King's English in the face of the Mother Country as a demonstration of disrespect and independence. Two hundred years later, we're still at it, continually exhibiting our disdain for convention and authority.

But more importantly, America has found itself at the forefront of some of the most significant developments in recorded Western history. And as each new situation has arisen, we have struggled to find a new vocabulary to describe the uniqueness of our experiences. Culture, art, science, politics, psychology, religion —each change, each evolutionary and revolutionary step has inspired a new set of words, a new range of self-expression.

Which brings us to the subject at hand: computers.

This era of the computer offers word hounds the opportunity to study firsthand the impact a new technology can have on language. Through a process of theft, invention and creative redefinition, the computer world has developed an elaborate and highly specialized lexicon.

Computerists have a basketful of new and borrowed words at their disposal. Examples include "modem," "software," "bit" (short for "*b*inary dig*it*"), "access" (as a verb), "peripheral" (as a noun), "data" (as a collective noun meaning "information"), "COBOL," "BASIC," "FORTRAN," "byte" and the verbs "peek" and "poke."

Fortunately for the language—and unfortunately for the editors who have to make some sense out of it—most computerists are blissfully ignorant of such details as consistency, spelling and proper usage. One fellow writes "BASIC," and another writes "Basic"; one prefers "disk" while another uses "disc"; one says "data is" and another says "data are."

It is time, however, to decide on a lingua franca. While foolish consistency may be the hobgoblin of little minds, wise consistency is the friend of those who wish to communicate concisely and effectively.

This is where editors enter the picture. It is our job to make the pronouncements that no one else has the temerity to make.

This doesn't mean that we make any actual decisions. Language is the property of the people, and history has proven that authorities and experts are curtly dismissed from the table. (Even Noah Webster couldn't popularize such inventions as "ile" for "aisle" and "ake" for "ache.")

But we can keep track of what's going on. And in the process, perhaps we can make a few more people aware of their language and how it should be used.

This column, then, will be a scorecard of the computerist's language—what it is, where it comes from, and where it is going. I'll cover such topics as the origin and history of words, the infusion of computer jargon into the common vernacular, capitalization and acronyms, word invention and word meaning.

In addition, I'll be citing examples from the pages of this and other magazines to illustrate points or raise questions.

Have any comments? Know any etymological trivia? (I'd love to know, for example, the origin of the word "snivitz.") Feel free to write. Perhaps in the coming months we can all learn to use our communications skills a little more effectively.

* * * * *

"Third," "second" and "first" are the ordinal correspondents to the cardinal numbers "three," "two" and "one." So what's the ordinal partner of "zero"? Several of our writers have suggested "zeroeth," as in the phrase "on the zeroeth track of the disk." This innovation, while pragmatic and practical, has not gone over big here at the magazine. I prefer "naughth," which would seem to have a cousin in the word "eighth," but nobody likes this one, either. Our final decision: the ordinal equivalent of "zero" will be "zero." And another golden opportunity to get into the *Oxford English Dictionary* goes by the boards.

Edited by Eric Maloney

Banking At Home

Service could presage big boost for Radio Shack

Micros and Banking

Express Information, a home banking service that debuted recently at the United American Bank in Knoxville, TN, could signal an important new microcomputing marketing strategy for the 80s.

By next year, Radio Shack could place as many as 100,000 new TRS-80 Color Computers in the homes of bank customers across the country who want to take advantage of the United American Service Corporation's new bank-by-computer product. In addition, customers will have access to the CompuServe Information Service, significantly increasing its total number of subscribers.

While no one is making any predictions, the service—developed and marketed to banks by the UASC—could spread rapidly. The UASC has already been contacted by some 400 banks, including, according to one report, five of the top 50 in the nation.

The UASC offers data processing and marketing services for more than 30 financial institutions in Tennessee, Virginia and Kentucky. It is owned by 12 southeastern banks, including the UAB.

Because the UAB service started only last fall, bold forecasts are risky. But the UASC, Radio Shack and CompuServe are confident that the hardware and software capabilities exist. The largest deterrent at the moment seems to be how successful the American banking industry perceives the Knoxville efforts. Traditionally, bankers do not display the microcomputer industry's bubbly entrepreneurial optimism.

How the Service Works

The way the service works, the bank issues a certificate to the new home banking customer. He takes it to a Radio Shack store, which provides him with the computer and a modem. Customers can either lease or buy the computer.

The UAB's first group of microcomputer bankers—they number about 400—are being offered limited service. But by the end of 1981, they will be able to use their micros to pay bills, to monitor checking account transactions and for bookkeeping.

A customer could, for example, instruct the bank to pay his electric bill, specifying the amount and the date. In the process, he could get a readout of all his previous transactions and his current balance.

Customers will also be able to use the service's file-sorting capabilities at tax time, calling up, for instance, a record of all medical expenses for the year.

The success of the Express Information service hinges largely on consumer acceptance of banking without the comfort of checks, bank books and hard cash. Tom Sudman, UASC president and UAB executive vice-president, says that the UASC has developed a number of ways to make the customer feel that his checking account is secure and reliable.

"All the resistance has come when we do not give the consumer reinforcement," he says. "But we are now seeing enthusiasm we did not have before because people have seen the reinforcement they're getting."

When the customer wants to pay a bill, for example, the computer asks a series of questions so the customer knows exactly how much money he is paying, where it is going and when. The system includes options to cancel, and transactions are reversible.

Also, at the end of the month, each customer receives a hard copy of his transactions, which can be checked against the computer's and customer's records.

Sudman also acknowledged customer concerns that their accounts might be

✓ Reader Service—see page 226

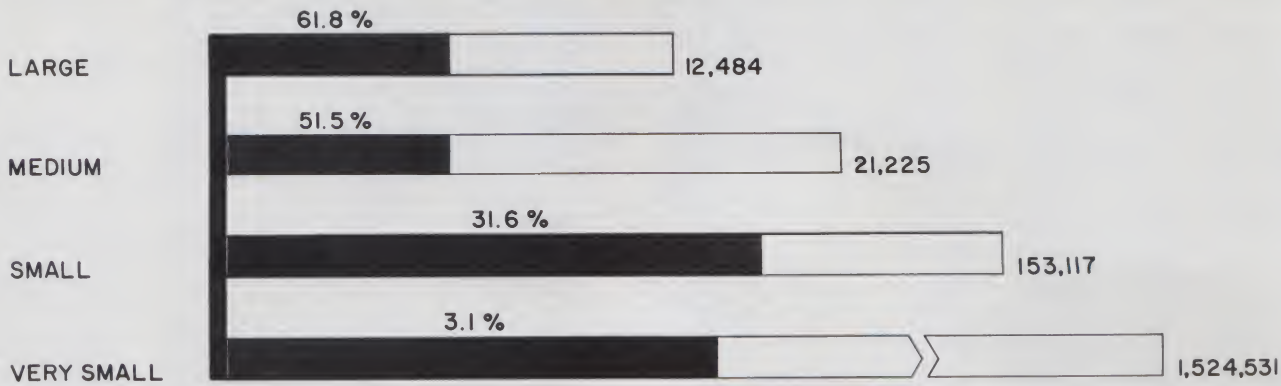


Table 1. Computer penetration by company size. While 62 percent of the larger companies, vs only 32 percent of the small and three percent of the very small companies, have systems, the actual number of installations is about equal. (In this chart, latter numbers represent study's projected, not total, universe.) (Source: Focus Research Systems, Inc.)

tampered with by sophisticated computer crooks. The bank hopes to provide adequate security with a special ROM security pack and a secret code number for each customer.

"I can't say that it's never going to happen," Sudman says. "But when everything is simply entered in ledgers in the back of the bank, someone on the cleaning crew could look at your records and we'd never know. At least here we know it's happened and can trace it down." He added that the UAB has successfully traced and prosecuted every case in which it has been involved.

Other Electronic Services

Express Information is only one of a series of electronic banking services being offered by the UAB.

"The banking industry first moved to full-service, then to drive-ins, and then to branch banking," says Sudman. "Today it's 24-hour automatic tellers. But we're moving into a new hierarchy of products."

As part of its electronic service, the UAB has built four Express Banking Centers in outlying districts of Knoxville. The buildings are completely electronic, and offer 85 percent of the bank's services.

The bank also offers an Express Statement service, which lets the customer get summary statements on a CRT terminal in the bank lobby. The customer can also push a button and get a hard copy printout.

And a service called Express Tel-a-Pay lets customers pay bills over the telephone.

Convenience, of course, is the major advantage of electronic banking. But the bank stands to benefit, too. It costs 40–50 cents to process a check; the same item can be moved electronically for about 3 cents. And fewer people are needed to handle paperwork.

A Look Ahead

All the parties involved acknowledge that home computer banking is not an end in itself.

"We want the consumer to develop a relationship with their Radio Shack store," says Sudman. "We're trying to convince the consumers that while they might not be interested in personal or home computing, they might be interested in better financial personal management."

CompuServe's Richard Baker echoes Sudman's remarks: "We're always interested in extending the scope of the CompuServe Information Service, and we see this as an opportunity to do that. We also recognize that banking is a very popular way of getting into home computers. Everybody can identify with banking; everybody banks in one form or another."

In other words, banking by microcomputer offers a legitimate practical application for the average consumer. Once he's got his TRS-80 installed, he can be led to discover CompuServe and other micro capabilities.

How fast could it spread, and to what degree? Sudman says 20,000 installations by from five to 20 banks by the end of this year. David Beckman, vice-president of advertising at Tandy, says that 20,000 is a low-end estimation: it could be as many as 100,000. In any case, the UASB is not restricted to the 30-odd banks it serves; nothing is holding it back from expanding nationwide, which is their expressed intent.

"It wouldn't take too much to offer this service nationwide," says Baker. "The software exists; it's a question of hardware."

"We've already manufactured a quarter of a million computers with little trouble," says Beckman. "We can supply them as they need them."

Does the Knoxville project presage an era of completely electronic banking? Sudman says no.

"One of the mistakes bankers make is saying it's an either-or proposition. But consumers enjoy having a choice. This is not going to mean the end of checks. People will still write checks; they will still put money under their mattresses."

This is the first of two parts on electronic banking. Part two will appear in next month's Micro-Scope column.

Small Businesses Buying Systems

Only 6.7 percent of American small businesses have on-site business computers, says a recent market study, but that figure could increase by up to 25 percent within 12 months.

The study, "Small Computers for Small Business," was done by Focus Research Systems, Inc., for Time Magazine. It is based on a larger study, "Small Businesses: Computing and Data Processing," published earlier last year.

The study defined "small business" as any company with less than 500 employees or with less than $25 million in sales. According to Census Bureau statistics, some 3.5 million companies qualify as small businesses. Based on its original sample of some 20,000 businesses, Focus Research estimates that some 195,000 companies have computers. The vast majority—nearly 82 percent—use manual methods to process data.

When broken down by company size, Focus's statistics show that computer penetration is most with larger companies and least with the smaller companies. (See Table 1.) Conversely, medium and small companies are more likely to buy systems.

Following are some other statistics compiled by FRS:

- Of the types of businesses surveyed by Focus, manufacturing has the highest computer penetration, with 9.8 percent, followed by wholesale businesses (8.6 percent) and transportation, communication and utilities (8.6 percent). At the bottom of the list are finance (3.4 percent) and agriculture (4.5 percent). (See Table 2.)

●Accounts receivable, accounts payable and payroll lead the list of most frequently mentioned applications, while sales analysis, cost accounting and order entry are at the bottom. But among possible users, inventory control is first. (See Table 3.)

"Regardless of size or industry, they want computers principally for traditional 'bread and butter' applications," says the Time Magazine report, while the original study adds, "None of the highly touted new applications—word processing, for example—had any standing."

●In the original study, 60 percent of those companies questioned had no staff programmer.

●International Business Machines leads the pack of vendors who have installed computer systems in small businesses, with a 33.8 percent share. It is followed by Burroughs (10.2 percent), NCR (7 percent) and DEC (5.9 percent).

Focus defined a small business computer as any computer "found to be used in small businesses."

For more information on the Time Magazine study, write Mr. John H. Stradal, Time Marketing Department, Time and Life Building, Rockefeller Center, New York, NY 10020. The detailed study is available for $495 from Focus Research Systems, Inc., 342 North Main St., West Hartford, CT 06117 (203-561-1047).

Type of Business	Pct.
Agriculture	4.5
Mining and Const.	5.6
Mfg.—Process	10.6
Mfg.—Discrete	9.4
Trans., Comm., Util.	8.6
Wholesale—Durable	8.6
Wholesale—Nondurable	8.6
Retail	6.6
Finance	3.4
Services	5.5

Table 2. Computer penetration by type of company. (Source: Focus Research Systems, Inc.)

Current Users	Pct.
Accounts Receivable	51
Accounts Payable	45
Payroll	40
General Ledger	26
Inventory Control	26
Billing	23
Purchasing	15
Order Entry	8
Cost Accounting	7
Sales Analysis	6
Possible Purchasers	**Pct.**
Inventory Control	42
Accounts Receivable	34
Accounts Payable	28
Payroll	23
Billing	15
General Ledger	14
Purchasing	10
Cost Accounting	8

Table 3. Most frequently mentioned computer applications. Current users listed their three most important applications, while possible users mentioned their two most important applications. (Source: Focus Research Systems, Inc.)

FCC Turns to Darwin

The Federal Communications Commission, says Chairman Charles Ferris, has entered a new era of regulatory Darwinism: it now insures the survival of the fittest instead of the survival of the fattest.

Ferris, speaking before the National Press Club in Washington last October, said that the FCC "is allowing new technologies and new entrepreneurs to achieve the potential that burdensome regulation denied them in a previous era."

The FCC, Ferris pointed out, has, within the last three years, opened up the common carrier market, deregulated satellite earth stations, ended Western Union's telegram monopoly and AT&T's long-distance telephone monopoly, created new clear channel assignments for radio, increased the number of VHF television stations around the country and deregulated cable and subscription television.

These policies, he said, "are the essence and the genius of the free enterprise system: that the public and not the government can have the power to choose the product they want at a price they are willing to pay. Through their choices, some entrepreneurs will win and some will lose, but they all must try harder."

Ferris compared the telecommunications and information industries of today with the steel and automobile industries of ten years ago, and warned that the former must anticipate the problems that led to the current crises facing the latter.

"I am determined that we not allow an environment that impedes innovations in our information services and allows others to surpass this country's leadership in the fields of telecommunications and information handling," he said.

Ferris pointed in particular to the First Amendment rights of free speech guaranteed to the print media, but which many people claim are denied the broadcast media. Television sets will soon become receivers of information traditionally found in newspapers, and "how we treat the legal obligations of that information tomorrow may well depend upon the broadcasting framework we create today."

Will new technologies thrive at the expense of older ones? No, Ferris said. The FCC is simply developing market structures that "exact a price from those who are complacent." It is up to the television networks, the telephone companies and the print media to meet the challenges of the future.

"I issue these challenges—to the established industries looking for a way to stay on top, and to the new entrepreneurs seeking a foothold—in a spirit of faith and optimism," Ferris concluded. "I know that the networks, the telephone companies and our nation's newspapers and magazines hold vast and as yet still untapped potential for serving the public."

Computer Tutors for Delinquents

An organization in Brooklyn, NY, is using computer games as part of its tutoring program for former offenders and youths with learning problems.

An article in the November 2 issue of *The New York Times* describes how the Fortune Society uses such games as tick-tack-toe, hangman and concentration to stimulate the interest of youths who might otherwise be on the streets raising hell. The Society uses a Commodore PET.

The idea behind the program is that while some kids are not interested in math per se, they might be interested in computer games for which they must learn math.

Divorced? Tired? Alone? Join the Club

Are you a divorced, tired loner who prefers machines over people? Then you've got the characteristics of the average data processing professional, says a recent issue of *Datamation* magazine.

The report says that the unconventional hours, the deadlines and the stress cause professionals to have trouble communicating with people and resolving conflicts. As a result, the person's friends, family and fellow employees also become more stressed.

The report says these problems will be partly alleviated as the computer industry—particularly software manufacturing—stabilizes.

MICRO QUIZ

What Does This Program Do?

What integer will be printed when the following program is executed?

```
S=0
N=48
FOR D=2 TO INT (N/2)
    IF N/D=INT (N/D) THEN S=S+1
NEXT D
PRINT S
```

(answer on page 204)

LETTERS TO THE EDITOR

OSI in the sky, Supporting the 6800, In defense of Heath, TRS-80 to the rescue

Out of the Blue

"OSI in the Sky" (October 1980, p. 102) is *exactly* the kind of how-we-did-it, what-it-cost article I'd like to see more of in the future.

Enthusiastic computer salesmen convince too many small-business owners that an expenditure of less than $2000 can solve all their problems overnight. Having articles like this from Sky Publishing enables us unbiased bystanders to counter some of the salesmen's claims. While subscription fulfillment is not the average business problem, the article points out the costs and time required to properly implement the solution to a problem of a certain size.

Surely more readers could provide details of some of their successes and failures in attacking the problems of the small-business operator. Let's see more articles like this, please!

Ken Barbier
Borrego Springs, CA

The Cost of a Utility

The article by Frank Derfler ("What Is the Utility of a Utility?" October, p. 72) suggests that "You can buy a lot of five-dollar hours for all of the money you might spend on disk drives, memory and languages" by connecting to a computer network like The Source or MicroNET. Whether this is true or not depends on the number of connect hours you use and if you make use of "non-prime" time. Small businesses often must use computer services during prime time (6 AM–6 PM, Monday–Friday) and only rarely require highly sophisticated software unavailable for microcomputers. If only four hours of connect time are used each workday, The Source charges about $1200 monthly. In less than one year it would cost less to purchase any of several business-oriented microcomputers and the best software now available.

The Source or MicroNET does offer rapid access to interesting information, but small businesses are well advised to carefully investigate the expense of using such computer networks for all their computer needs.

Richard Ainsley
Visalia, CA

Support for the 6800

John Travares' letter in the October issue ("SWTP Is No Fun," p. 27) makes some excellent points. I, too, own an SWTP 6800 and am very pleased with it. My complaint is that every time I turn around, someone is attempting to sell me 6809 hardware and/or software. Now, I am familiar with the 6809 and am well aware of its advantages over the 6800. However, not all of us can afford to upgrade our systems to 6809s.

I guess my real hang-up is that when I write to a company and specify that I want information on products for the 6800, I expect at least half of the information to be on the 6800. Instead, at least 90 percent, and usually 100 percent, of the information I receive is for the 6809. Isn't anyone going to support the 6800 anymore?

Ken Fulton
Morgantown, WV

Super Snooper

Ken Barbier's article on the 8085's undocumented op codes ("The Secret Life of the 8085," September, p. 114) shows a good deal of careful detective work. He raises a number of points in his article. The RHR (rotate HL right through carry) instruction leaves the high bit unchanged, but far from being "more trouble to learn than it is worth," this instruction can be very useful. Anyone who has done a moderate amount of 16-bit programming will recognize this instruction as an arithmetic shift right, aka divide by two on a signed 16-bit integer.

The caveats at the end of the article *cannot* be overstressed. These mnemonics will work only with 8085s from Intel (or from another vendor using Intel masks) and only until Intel decides to make design changes to the chip. As Ken said: "Be forewarned." But forewarned is forearmed, and 16-bit instructions are handy (even if you didn't pay for 'em!).

H. W. Neff
San Leandro, CA

Leave Your Options Open

Re "Computerized Estate Planning" (*Kilobaud Microcomputing*, October 1980, p. 31).

The subtitle, "sorting the various options for the final settlement," is rather ambitious, since the program data consists entirely of factual data, except perhaps the "estimated or assumed year of the wife's death."

The program makes some unstated assumptions, such as that the income from savings prior to the man's death is spent, not compounded. This is unlikely when data item 890 is not zero.

There is no provision for savings other than cash. In inflationary times, real estate investments, more or less like the home, would behave reasonably predictably.

It would seem that the interest rate on the home would better be a data item. The same holds for the information in line 122, which would not then have to be optional.

It would be useful to be able to print the inner loop, and perhaps the outer loop also, at small intervals, for example, five years.

(continued on page 227)

How's your love life?

A little dull around the edges? Routine? Predictable? Boring? Maybe all it needs is a little Interlude. Interlude is the most stimulating computer game ever conceived. It combines a computer interview, an innovative programming concept, and a one-of-a-kind manual to turn your love life into exciting, adventurous, delicious fun!

Interlude is: romantic... playful... outrageous... a fantasy. Interlude is: ■ A Bed of Roses (Interlude #1) ■ Mata Hari (Interlude #49) ■ The Chase (Interlude #7) ■ Rodeo! (Interlude #71) ■ The King and I (Interlude #60) ■ Some Enchanted Evening (Interlude #84) ■ Caveman Caper (Interlude #82) ■ From Here to Ecstasy (Interlude No. 30) ■ Satin Dreams (Interlude #72).

More than 100 Interludes are included in the program. Most are described in detail in the accompanying manual, but several surprise Interludes are buried in the program awaiting that very special time when your interview says you're ready. (When you learn secret Interlude #99, your love life may never again be the same!) Interlude can give you experiences you'll never forget. Are you ready for it?

Interlude™ ✓235
The Ultimate Experience.

INTERLUDE, 10428 Westpark, Houston, Texas 77042. I'm really ready. Send my Interlude today.

Apple II (16K)*
☐ Cassette ($16.95)
☐ Diskette ($19.95)
☐ Diskette—Pascal or DOS 3.3 ($19.95)
Add $1.50 for shipping and handling.

TRS-80 (Level II-16K)**
☐ Cassette ($16.95)
☐ Diskette ($19.95)

Poster
☐ 20"x 24" reproduction of this ad without ad copy ($4.95—includes shipping charges)

Available for immediate shipment.

Please enclose your check payable to INTERLUDE or complete the charge information:

☐ MASTERCARD ☐ VISA All charge customers must sign here. ____________
Account No. ____________ Expiration date ________ MasterCard Bank Code ________

CHARGE CUSTOMERS: Order by phone toll-free! **1-800-231-5768 Ext. 306** (Texas: 1-800-392-2348 Ext. 306)
Name ____________ Age ______
Address ____________
City ____________ State ________ Zip ________

*Apple II is a registered trademark of Apple Computers, Inc. **TRS-80 is a registered trademark of Radio Shack, a Tandy Co.

No.18

Software with full support

Purchasing our software is just the beginning. We then back it up with professional support:

■Subscription to "LIFELINES" for automatic notifications of revisions! ■Update service for software and documentation! ■Telephone Hotline! ■Overseas software export service!

All Lifeboat programs require CP/M, unless otherwise stated.

Software with Manual / Manual Alone

☐ **CP/M® FLOPPY DISK OPERATING SYSTEM**—Digital Research's operating system configured for many popular micro-computers and disk systems:

System	Version	Price
Apple II® SoftCard® with Z80 Microsoft BASIC version 5 with high resolution graphics	2.x	349/NA ◇
North Star Single Density	2.x	170/25 Ⓡ
North Star Double/Quad	2.x	170/25 Ⓡ
Durango F-85	2.x	170/25
iCOM Micro-Disk 2411	1.4	145/25
iCOM 3712 for MITS 88-2SIO Console	1.4	170/25 *
iCOM 3712 for 3P + S/MITS SIO Rev non-zero console	1.4	170/25 *
iCOM 3812	2.x	225/25 Ⓡ
iCOM 3812	1.4	170/25 *
iCOM 4511/Pertec D3000	2.x	375/25 *+
Mits 3202/Altair 8800	1.4	145/25
Heath H8 + H17	1.4	145/25 Ⓜ
Heath H89 by Magnolia	2.x	249/25 ◇
Ohio Scientific C3	2.x	200/25
Ohio Scientific C3-C	2.x	250/25
Onyx C8001 Standard	2.x	250/25
Onyx C8001 Enhanced	2.x	330/25
TRS-80 Model I	1.4	145/25 Ⓜ
TRS-80 Model II	2.x	170/25
TRS-80 Model II + Corvus	2.x	250/25
Processor Technology Helios II	1.4	145/25
Intel MDS Single Density	2.x	170/25
Intel MDS Double Density	2.x	170/25
New→ Micropolis Mod I	2.x	200/25 Ⓡ
New→ Micropolis Mod II	2.x	200/25 Ⓡ
Mostek MDX STD Bus System	2.x	350/25 **

HARD DISK OPTIONS (items with Ⓡ) when purchased with CP/M 2.X system.

Corvus	**Add 80.**
Konan Phoenix	**Add 80.**
iCOM 4511/Pertec D3000	**Add 80.**

*Software consists of the operating system, text editor, assembler, debugger and other utilities for file management and system maintenance. Complete set of Digital Research's documentation and additional implementation notes included. Systems marked * and ** include firmware on 2708 and 2716. Systems marked + include 5440 media charge. Systems marked Ⓜ require the special Ⓜ versions of software in this catalog. ◇ includes hardware addition to allow our standard versions of software to run under it.*

☐ **Z80 DEVELOPMENT PACKAGE**—Consists Ⓜ ② of: (1) disk file line editor, with global inter and intra-line facilities; (2) Z80 relocating assembler, Zilog/Mostek mnemonics, conditional assembly and cross reference table capabilities; (3) linking loader producing absolute Intel hex disk file .... **$95/$20**

☐ **ZDT**—Z80 Monitor Debugger to break and Ⓜ ② examine registers with standard Zilog/Mostek mnemonic disassembly displays. $35 when ordered with Z80 Development Package .... **$50/$10**

AVOCET SYSTEMS

☐ **XASM-68**—Non-macro cross-assembler with nested conditionals and full range of pseudo operations. Assembles from standard Motorola MC6800 mnemonics to Intel hex ... **$200/$25**

☐ **XASM-65**—As XASM-68 for MOS Technology MCS-6500 series mnemonics ...... **$200/$25**

☐ **XASM-48**—As XASM-68 for Intel MCS-48 and UPI-41 families .... **$200/$25**

☐ **XASM-18**—As XASM-68 for RCA 1802 .... **$200/$25**

☐ **DISTEL**—Disk based disassembler to Intel 8080 or TDL/Xitan Z80 source code, listing and cross reference files, Intel or TDL/Xitan pseudo ops optional. Runs on 8080 .... **$65/$10**

☐ **DISILOG**—As DISTEL to Zilog/Mostek Ⓜ ② mnemonic files .... **$65/$10**

☐ **SMAL/80** Structured Macro Assembler Ⓧ Language—Package of powerful general purpose text macro processor and SMAL structured language compiler. SMAL is an assembler language with IF-THEN-ELSE, LOOP-REPEAT-WHILE, DO-END, BEGIN-END constructs. Not compatible with CP/M version 2 or greater. .... **$75/$15**

Software with Manual / Manual Alone

☐ **RAID**—8080/8085 debugging aid and general system utility program. Both a software emulator and real-time debug monitor. Features multiple breakpoints, symbolic input, symbolic display and altering of registers, built-in assembler & dis-assembler, tracing, single-stepping, memory protection, histograms, memory search, direct disk access. Requires 32K CP/M .... **$250/$25**

PHOENIX SOFTWARE ASSOCIATES

☐ **PASM®**—Z80 macro assembler, Intel/TDL ② mnemonics. Generates Intel hex format or relocatable code in either TDL Object Module format or PSA Relocatable Binary Module format. Supports text insertion, conditional branching within macros, recursive macro calls and parameter passing. .... **$129/$25**

☐ **EDIT**—Character oriented text file editor. Includes ② macro definition capabilities. Handles insertion, deletion, searching, block move, etc. for files of any length. Does not require a CRT. .... **$129/$25**

☐ **PLINK®**—Two pass **disk-to-disk linkage editor/loader** ② which can produce re-entrant, ROMable code. Can link programs that are larger than available memory for execution targeted on another machine. Full library capabilities. Input can be PSA Relocatable Binary Module, TDL Object Module or **Microsoft REL files**. Output can be a COM file, Intel hex file, TDL Object Module or PSA Relocatable file. .... **$129/$25**

☐ **BUG® and μBUG®**—Z80 interactive machine ② level debugging tools for program development. BUG has full mnemonic trace and interactive assembly (mnemonics compatible with PASM). Dynamic breakpoints and conditional traps while tracing (even through ROM!). μBUG is a subset of BUG and is used in memory limited situations .... **$129/$25**

DIGITAL RESEARCH

☐ **MP/M**—Installed for single density MDS-800. Multi-processing derivative of the CP/M operating system. Manual includes CP/M2 documentation .... **$300/$50**

☐ **MAC**—8080 Macro assembler. Full Intel macro ⑤ definitions. Pseudo Ops include RPC, IRP, REPT, TITLE, PAGE, and MACLIB. Produces absolute hex output plus symbol table file for use by SID and ZSID (see below) . **$120/$15**

☐ **SID**—8080 Symbolic debugger. Full trace, ⑤ pass count and breakpoint program testing. Has backtrace and histogram utilities. When used with MAC, provides full symbolic display of memory labels and equated values . **$105/$15**

☐ **ZSID**—Z80 Symbolic debugger with all features ⑤ ② of SID .... **$130/$15**

☐ **TEX**—Text output formatter to create paginated, ⑤ page-numbered and justified copy. Output can be directed to printer or disk ... **$105/$15**

☐ **DESPOOL**—Utility program to permit simultaneous ⑤ printing from text files while executing other programs .... **$80/$10**

☐ **tiny C**—Interactive interpretive system for Ⓧ teaching structured programming techniques. Manual includes full source listings . **$105/$50**

☐ **BDS C COMPILER**—Supports structures, Ⓜ ① unions, 2 dimensional arrays, pointers, recursion and overlays. Features optimized code generator, variable sized buffers for file I/O, and capability to produce ROMable code. Includes macro package to enable user to produce linkable modules with MAC (see under Digital Research). Floating point functions, full run-time package and machine code library sources provided. Linker, library manager and textbook included. Compiler lacks initializers, statics, floats and longs. .... **$145/$25**

☐ **WHITESMITHS C COMPILER**—The ultimate Ⓛ Ⓧ in systems software tools. Produces faster code than a pseudo-code Pascal with more extensive facilities. Conforms to the full UNIX® Version 7 C language, described by Kernighan and Ritchie, and makes available over 75 functions for performing I/O, string manipulation and storage allocation. Linkable to Microsoft REL files. Requires 60K CP/M ..... **$630/$30**

MICROSOFT

Software with Manual / Manual Alone

☐ **BASIC-80**—Disk Extended BASIC, ANSI Ⓛ Ⓜ compatible with long variable names, WHILE/WEND, chaining, variable length file records. MBASIC version 4.51 also included on disk. .... **$325/$25**

☐ **BASIC COMPILER**—Language compatible Ⓛ Ⓜ with BASIC-80 and 3-10 times faster execution. Produces standard Microsoft relocatable binary output. Includes MACRO-80. Also linkable to FORTRAN-80 or COBOL-80 code modules .... **$350/$25**

☐ **FORTRAN-80**—ANSI 66 (except for COMPLEX) Ⓛ Ⓜ plus many extensions. Includes relocatable object compiler, linking loader, library with manager. Also includes MACRO-80 (see below) .... **$425/$25**

☐ **COBOL-80**—Level 1 ANSI '74 standard plus Ⓛ Ⓜ most of Level 2. Full sequential, relative, and indexed file support with variable file names. Powerful interactive, formatted screen handling with ACCEPT and DISPLAY verbs. Program segmentation for execution of programs larger than memory and CHAIN command with parameter passing. Full support of CP/M version 2 files. Includes MACRO-80 (see above), linking loader, and relocatable library manager. Requires 48K CP/M .... **$700/$25**

☐ **M/SORT**—Optional sort/merge capability for Ⓛ COBOL-80 which conforms fully to SORT/MERGE, Level II of the 1974 ANSI COBOL standard (except COLLATING SEQUENCE IS alphabet-name). Requires COBOL-80. Sold as an update to COBOL-80 .... **$150/$10** New
COBOL-80 + M/SORT .... **$825/$35**

☐ **MACRO-80**—8080/Z80 Macro Assembler. Ⓛ Ⓜ Intel and Zilog mnemonics supported. Relocatable linkable output. Loader, Library Manager and Cross Reference List utilities included .... **$149/$15**

☐ **XMACRO-86**—8086 cross assembler. All Ⓛ Macro and utility features of MACRO-80 package. Mnemonics slightly modified from Intel ASM86. Compatibility data sheet available .... **$275/$25**

☐ **EDIT-80**—Very fast random access text editor Ⓛ for text with or without line numbers. Global and intra-line commands supported. File compare utility included. .... **$89/$15**

☐ **muSIMP/muMATH**—muSIMP is a high level Ⓛ programming language suitable for symbolic and semi-numerical processing implemented using a fast and efficient interpreter requiring only 7K bytes of machine code. muMATH is a package of programs written in muSIMP. The package performs sophisticated mathematical functions. Keeps track of up to 611 digits. Performs matrix operations on arrays: transpose, multiply, divide, inverse and other integer powers. Logarithmic, exponential, trigonometric simplification and transformation, symbolic differentiation with partial derivatives, symbolic integration of definite and indefinite integrals. Requires 40K CP/M .... **$250/$20**

☐ **muLISP-80**—Microcomputer implementation Ⓛ of LISP. The interpreter resides in only 7K bytes of memory yet includes 83 LISP functions. Has infinite precision integer arithmetic expressed in any radix from 2 to 36. muLISP80 includes complete trace facility and a library of useful functions and entertaining sample programs .... **$200/$15**

☐ **PASCAL/M®**—Compiles enhanced Standard Ⓧ Pascal to compressed efficient Pcode. Totally CP/M compatible. Random access files. Both 16 and 32-bit Integers. Runtime error recovery. Convenient STRINGs. OTHERWISE clause on CASE. Comprehensive manual (90 pp. indexed). SEGMENT provides overlay structure. INPORT, OUTPORT and untyped files for arbitrary I/O. Requires 56K CP/M. Specify 1) 8080 CP/M, 2) Z80 CP/M, or 3) Cromemco CDOS. .... **$175/$20**

☐ **PASCAL/Z**—Z80 native code PASCAL compiler. ① ② Produces optimized, ROMable re-entrant code. All interfacing to CP/M is through the support library. The package includes compiler, relocating assembler and linker, and source for all library modules. Variant records, strings and direct I/O are supported. Requires 56K CP/M .... **$395/$25**

☐ **PASCAL/MT**—Subset of standard PASCAL. Ⓜ Ⓧ Generates ROMable 8080 machine code. Symbolic debugger included. Supports interrupt procedures, CP/M file I/O and assembly language interface. Real variables can be BCD, software floating point, or AMD 9511 hardware floating point. Includes strings enumerations and record data types. Manual explains BASIC-PASCAL conversion. Requires 32K . **$250/$30**

☐ **APL/V80**—Concise and powerful language for ② application software development. Complex programming problems are reduced to simple expressions in APL. Features include up to 27K active workspace, shared variables, arrays of up to 8 dimensions, disk workspace and copy object library. The system also supports auxiliary processors for interfacing I/O ports. Requires 48K CP/M and serial APL printing terminal or CRT .... **$500/$30**

☐ **ALGOL-60**—Powerful block-structured language ② compiler featuring economical run-time dynamic allocation of memory. Very compact (24K total RAM) system implementing almost all Algol 60 report features plus many powerful extensions including string handling direct disk address I/O etc. .... **$199/$20**

☐ **CBASIC-2** Disk Extended BASIC—Non-interactive Ⓜ BASIC with pseudo-code compiler and run-time interpreter. Supports full file control, chaining, integer and extended precision variables, etc. Versions of CRUN for CP/M versions 1.4 and 2.x included on disk. . **$120/$15**

MICRO FOCUS

Software with Manual / Manual Alone

☐ **STANDARD CIS COBOL**—ANSI '74 COBOL Ⓛ standard compiler fully validated by U.S. Navy tests to ANSI level 1. Supports many features to level 2 including dynamic loading of COBOL modules and a full ISAM file facility. Also, program segmentation, interactive debug and powerful interactive extensions to support protected and unprotected CRT screen formatting from COBOL programs used with any dumb terminal .... **$850/$50**

☐ **FORMS 2**—CRT screen editor. Output is Ⓛ COBOL data descriptions for copying into CIS COBOL programs. Automatically creates a query and update program of indexed files using CRT protected and unprotected screen formats. No programming experience needed. Output program directly compiled by STANDARD CIS COBOL .... **$200/$20**

☐ **NEVADA COBOL**—Subset of ANSI-74. Features Ⓜ fast compilation and execution with small object modules. Has extended arithmetic with 18 digit accuracy. Extended I/O includes random access files and sequential files of both fixed and variable length records, and interactive accept/display verbs. Good error messages and debugging facilities enhance program development. Requires a 32K CP/M system .... **$149/$25** New

EIDOS SYSTEMS

☐ **KBASIC**—Microsoft Disk Extended BASIC Ⓛ version 4.51 integrated with KISS Multi-Keyed Index Sequential and Direct Access file management as 9 additional BASIC commands. KISS included as relocatable modules linkable to FORTRAN-80, COBOL-80, and BASIC COMPILER. Specify CP/M version 1.4 or 2.x when ordering. Requires 48K CP/M **$585/$45**
To licensed users of Microsoft BASIC-80 (MBASIC) .... **$435/$45**

☐ **XYBASIC** Interactive Process Control BASIC—Full disk BASIC features plus unique commands to handle byte rotate and shift and to test and set bits. Available in several versions:

Integer ROM squared	**$350/$25**
Integer CP/M	**$350/$25**
Extended ROM squared	**$450/$25**
Extended CP/M	**$450/$25**
Extended Disk CP/M	**$550/$25**
Integer CP/M Run Time Compiler	**$350/$25**
Extended CP/M Run Time Compiler	**$450/$25**

☐ **RECLAIM**—A utility to validate media under CP/M. Program tests a diskette or hard disk surface for errors, reserving the imperfections in invisible files, and permitting continued usage of the remainder. Essential for any hard disk. Requires CP/M version 2. ...... **$80/$5**

☐ **BASIC UTILITY DISK**—Consists of: (1) Ⓜ CRUNCH-14—Compacting utility to reduce the size and increase the speed of programs in Microsoft BASIC 4.51, BASIC-80 and TRS-80 BASIC. (2) DPFUN—Double precision subroutines for computing nineteen transcendental functions including square root, natural log, log base 10, sine, arc sine, hyperbolic sine, hyperbolic arc sine, etc. Furnished in source on diskette and documentation .... **$50/$35**

☐ **STRING/80**—Character string handling plus routines for direct CP/M BDOS calls from FORTRAN and other compatible Microsoft languages. The utility library contains routines that enable programs to chain to a COM file, retrieve command line parameters and search file directories with full wild card facilities. Supplied as linkable modules in Microsoft format. **$95/$20**

☐ **STRING/80** source code available separately— **$295/NA**

☐ **THE STRING BIT**—FORTRAN character Ⓜ string handling. Routines to find, fill, pack, move, separate, concatenate and compare character strings. This package completely eliminates the problems associated with character string handling in FORTRAN. Supplied with source .... **$65/$15**

☐ **VSORT**—Versatile sort/merge system for fixed Ⓜ length records with fixed or variable length fields. VSORT can be used as a stand-alone package or loaded and called as a subroutine from CBASIC-2. When used as a subroutine, VSORT maximizes the use of buffer space by saving the TPA on disk and restoring it on completion of sorting. Records may be up to 255 bytes long with a maximum of 5 fields. Upper/lower case translation and numeric fields supported. .... **$175/$20**

☐ **IBM/CPM**—Program to transfer IBM 3741 data set files to CP/M files or CP/M files to IBM 3741 data sets. Easy to use. Requires two eight inch diskette drives, 24K memory, and a 24 by 80 CRT terminal .... **$175/$5** New

CPAids®

☐ **MASTER TAX**—Professional tax preparation Ⓛ † program. Prepares schedules A, B, C, D, E, F, G, R/RP, SE, TC, ES and forms 2106, 2119, 2210, 3468, 3903, 2441, 4625, 4726, 4797, 4972, 5695 and 6251. Printing can be on readily available, pre-printed continuous forms, on overlays, or on computer generated, IRS approved forms. Maintains client history files and is interactive with CPAids GENERAL LEDGER II (see below) .... **$995/$30**
Annual Update Fee **$350**

☐ **STANDARD TAX**—As above for schedules A, † B, C, D, E, G, R/RP, SE, TC and forms 2106 and 2441. Also, does not maintain client history files .... **$495/$30**
Annual Update Fee **$175**

Neu in der Schweiz Lifeboat Associates GmbH, Aegeristr. 35, CH 6340 Baar **Telefon 042/31 2931**

Software with Manual / Manual Alone

☐ **GENERAL LEDGER II**—Designed for CPA's. † Stores complete 12 month detailed history of transactions. Generates financial statements, depreciation, loan amortizations, journals, trial balances, statements of changes in financial position, and compilation letters. Includes payroll system with automatic posting to general ledger. Prints payroll register, W2's and payroll checks .... **$450/$30**

☐ **T/MAKER**—Powerful new tool for preparing management reports with tabular data. Makes financial modeling projects easy. Do you want a weekly profitability report? Set up the table and compute. Just change the sales figures for next week and compute. You have a new report! T/MAKER includes a full screen editor for setting up tables which pages left, right, up and down. Compute includes standard arithmetic, percents, exponents, common transcendental functions, averages, maxima, minima, projections, etc. Requires 48K CP/M, CBASIC-2, CRT terminal with addressable cursor positioning. .... **$275/$25**

☐ **ESQ-1**—Professional time and billing for the Ⓛ legal profession. Designed for use by the first-† time computer user. Records billable and non-billable time. Complete system includes transaction entry, posting, billing, reports, and client analysis. Records cash receipts, escrow receipts, and escrow transfers. Requires 48K CP/M system, 480K of disk storage space, cursor addressable CRT, and CBASIC-2 .... **$1495/$50**

New

☐ Complete demonstration system for ESQ-1 Ⓛ † .... **$75/$50**

☐ **BSTAM**—Utility to link one computer to another Ⓜ also equipped with BSTAM. Allows file transfers at full data speed (no conversion to hex), with CRC block control check for very reliable error detection and automatic retry. We use it! It's great! Full wildcard expansion to send *.COM, etc. 9600 baud with wire. 300 baud with phone connection. **Both** ends need one. Standard and Ⓜ versions can talk to one another. This software requires a knowledge of assembler language for installation. .... **$150/$10**

☐ **BSTMS**—Intelligent terminal program for Ⓜ CP/M systems. Permits communication between micros and mainframes. Sends character data files to remote computers under complete control. System can record character data sent from remote computer systems and data banks. Includes programs to EXPAND and COMPRESS binary files for transmission. This software requires a knowledge of assembler language for installation. .... **$200/$25**

New

☐ **WHATSIT?***—Interactive data-base system using associative tags to retrieve information by subject. Hashing and random access used for fast response. Requires CBASIC-2 . **$175/$25**

☐ **SELECTOR III-C2**—Data Base Processor to † create and maintain multi-key data bases. Ⓜ Prints formatted sorted reports with numerical summaries or mailing labels. Comes with sample applications, including Sales Activity, Inventory, Payables, Receivables, Check Register, and Client/Patient Appointments, etc. Requires CBASIC-2. Supplied in source ..... **$295/$20**

☐ **GLECTOR**—General Ledger option to SELECTOR III-C2. Interactive system provides for customized COA. Unique chart of transaction types insure proper double entry bookkeeping. Generates balance sheets, P&L statements and journals. Two year record allows for statement of changes in financial position report. Supplied in source. Requires SELECTOR III-C2, CBASIC-2 and 56K system .... **$350/$25**

☐ **MAGSAM III**—Sophisticated keyed access file Ⓛ support system. Supports random, sequential, and generic retrieval by key. Also, multiple secondary keys. Dynamic allocation and extension of files with automatic free space reclamation. Interactive tutorial included to get the user started. Complete with documentation and quick reference card. Specify CBASIC or Microsoft BASIC version. Requires 48K system .... **$145/$25**

New

☐ **MAGSAM IV**—High speed machine code ver-Ⓛ sion of MAGSAM III for CBASIC only. Distributed as pre-loaded modules and Microsoft relocatable object modules .... **$295/$25**

New

DMA

☐ **CBS**—Configurable Business System is a † comprehensive set of programs for defining custom data files and application systems without using a programming language such as BASIC, FORTRAN, etc. Multiple key fields for each data file are supported. Set-up program customizes system to user's CRT and printer. Provides fast and easy interactive data entry and retrieval **with transaction processing**. Report generator program does complex calculations with stored and derived data, record selection with multiple criteria, and custom formats. Sample inventory and mailing list systems included. **No support language required** .... **$395/$40**

MICROPRO

☐ **SUPER-SORT I**—Sort, merge, extract utility as Ⓛ absolute executable program or linkable module in Microsoft format. Sorts fixed or variable records with data in binary, BCD, Packed Decimal, EBCDIC, ASCII, floating & fixed point, exponential, field justified, etc. Even variable number of fields per record! .... **$225/$25**

☐ **SUPER-SORT II**—Above available as abso-Ⓛ lute program only .... **$175/$25**

☐ **SUPER-SORT III**—As II without SELECT/Ⓛ EXCLUDE .... **$125/$25**

Software with Manual / Manual Alone

☐ **DATASTAR**—Professional forms control entry Ⓛ and display system for key-to-disk data capture. Menu driven with built-in learning aids. Input field verification by length, mask, attribute (i.e. upper case, lower case, numeric, auto-dup, etc.). Built-in arithmetic capabilities using keyed data, constant and derived values. Visual feedback for ease of forms design. Files compatible with CP/M-MP/M supported languages. Requires 32K CP/M and CRT with addressable cursor. .... **$350/$35**

☐ **WORD-STAR**—Menu driven visual word pro-Ⓛ cessing system for use with standard terminals. Text formatting performed on screen. Facilities for text paginate, page number, justify, center and underscore. User can print one document while simultaneously editing a second. Edit facilities include global search and replace, Read/Write to other text files, block move, etc. Requires CRT terminal with addressable cursor positioning .... **$445/$40**

☐ **WORD-STAR-MAIL-MERGE**—As above with Ⓛ option for production mailing of personalized documents with mail lists from DATASTAR or NAD .... **$575/$40**

☐ **WORD-MASTER** Text Editor—In one mode Ⓛ has superset of CP/M's ED commands including global searching and replacing, forwards and backwards in file in video mode, provides full screen editor for users with serial addressable-cursor terminal .... **$145/$25**

☐ **MAGIC WAND***—Word processing system with simple, easy to use full screen text editor and powerful print processor. Editor has all standard editing functions including text insert and delete, global search and replace, block move and library files for boiler plate text. Print processor formatting commands include automatic margins, pagination, headings & footings, centered and justified text. Also prints with true proportional spacing, merges with data files for automatic form letters, and performs run-time conditional testing for varied output. Requires 32K CP/M and CRT terminal with addressable cursor .... **$395/$40**

☐ **TEXTWRITER III**—Text formatter to justify and Ⓜ paginate letters and other documents. Special features include insertion of text during execution from other disk files or console, permitting recipe documents to be created from linked fragments on other files. Has facilities for sorted index, table of contents and footnote insertions. Ideal for contracts, manuals, etc. Now compatible with Electric Pencil* and Word-Star prepared files .... **$125/$20**

☐ **DATEBOOK**—Program to manage time just like an office appointment book but using the speed and memory of a computer. Keeps track of three appointment schedules (three dental chairs, three attorneys, etc.) at once. Appointments consist of name, reason for the appointment, the date and time, and the length of the appointment. System can be quickly customized for the individual user. Many helpful features for making, changing, finding, and reporting appointments. Requires 48K CP/M and 180K bytes diskette storage. Requires 80×24 cursor addressable terminal. Specify 8080 CP/M, Z80 CP/M or Cromemco CDOS .... **$295/$25**

New

PEACHTREE SOFTWARE*

☐ General accounting software for small busi-Ⓛ nesses. Each product can be used alone or with † automatic posting to the General Ledger. Supplied in source for Microsoft BASIC 4.51.

GENERAL LEDGER	$530/$40
ACCOUNTS PAYABLE	$530/$40
ACCOUNTS RECEIVABLE	$530/$40
PAYROLL	$530/$40
INVENTORY	$660/$40

☐ Other application products supplied in source Ⓛ for Microsoft BASIC 4.51.

† MAILING ADDRESS	$530/$40
PROPERTY MANAGEMENT	$925/$40

GRAHAM-DORIAN SOFTWARE SYSTEMS

☐ Comprehensive accounting software written in Ⓛ CBASIC-2 and supplied in source code. Each Ⓜ software package can be used as a stand-† alone system or integrated with the General Ledger for automatic posting to ledger accounts. Requires CBASIC-2.

GENERAL LEDGER	$805/$40
ACCOUNTS PAYABLE	$805/$40
ACCOUNTS RECEIVABLE	$805/$40
INVENTORY SYSTEM	$555/$40
JOB COSTING	$805/$40
APARTMENT MANAGEMENT	$805/$40
CASH REGISTER	$805/$40

☐ **POSTMASTER**—A comprehensive package Ⓜ for mail list maintenance that is completely menu driven. Features include keyed record extraction and label production. A form letter program is included which provides neat letters on single sheet or continuous forms. Includes NAD file translator. Requires CBASIC-2. .... **$150/$20**

STRUCTURED SYSTEMS GROUP

☐ Complete interactive accounting software for † business. Each product can be used standalone or with automatic posting to the general ledger. Each product is thoroughly tested and very well documented.

GENERAL LEDGER	$820/$40
ACCOUNTS RECEIVABLE	$820/$40
ACCOUNTS PAYABLE	$820/$40
PAYROLL	$820/$40
INVENTORY CONTROL	$820/$40

Software with Manual / Manual Alone

☐ **ANALYST**—Customized data entry and report-† ing system. User specifies up to 75 data items per record. Interactive data entry, retrieval, and update facility makes information management easy. Sophisticated report generator provides customized reports using selected records with multiple level breakpoints for summarization. Requires a disk sort utility such as QSORT, SUPER-SORT or VSORT and CBASIC-2 .... **$250/$15**

☐ **LETTERIGHT**—Program to create, edit and type letters or other documents. Has facilities to enter, display, delete and move text, with good video screen presentation. Integrates with NAD for form letter mailings. .... **$200/$25**

☐ **NAD**—Name and Address selection system. Interactive mail list creation and maintenance program with output as full reports with reference data or restricted information for mail labels. Transfer system for extraction and transfer of selected records to create new files. QSORT required if sorting is desired. **$100/$20**

☐ **QSORT**—Fast sort/merge program for files with fixed record length, variable field length information. Up to five ascending or descending keys. Full back-up of input files created .... **$100/$20**

★ ★ ★ ★ ★ ★ ★

☐ **HEAD CLEANING DISKETTE**—Cleans the drive Read/Write head in 30 seconds. Diskette absorbs loose oxide particles, fingerprints, and other foreign particles that might hinder the performance of the drive head. Lasts at least 3 months with daily use. Specify 5" or 8".
Single sided .... **$20 each/$55 for 3**
Double sided .... **$25 each/$65 for 3**

LIFELINES
NEWSLETTER FROM LIFEBOAT

LIFELINES is the first step in software support for the serious microcomputer user. Each issue reports new revisions together with information on the purpose for each such release, be it for correction of "bugs" or the addition of features and facilities.

Feature Articles ! New Software ! Product Comparisons ! Info on CP/M Users Group !

SUBSCRIPTION INFORMATION:

$18 for twelve issues: U.S., Canada, and Mexico.
$40 for twelve issues: all other countries.
$2.50 for each back issue: U.S., Canada, and Mexico.
$3.60 for each back issue: all other countries.

Send Check to LIFELINES, 1651 Third Avenue, New York, N.Y. 10028 or use your VISA or MASTERCARD—call (212) 722-1700

☐ **DC 300 Data Cartridges** Specify 450'XL or 300' certified. Pack of 5 .... **$100**

☐ **FLIPPY DISK KIT**—Template and instructions to modify single sided 5¼" diskettes for use of second side in single sided drives **$12.50**

☐ **FLOPPY SAVER**—Protection for center holes for 5" and 8" floppy disks. Only 1 needed per diskette. Kit contains centering post, pressure tool and tough 7 mil mylar reinforcing rings for 25 diskettes.
5", Kit **$14.95**
5", Rings only **$7.95**
8", Kit **$16.95**
8", Rings only **$8.95**

New

☐ **The CP/M HANDBOOK** (with MP/M) by Rodnay Zaks **$13.95**

☐ **PASCAL USER MANUAL AND REPORT**—By Jensen and Wirth. The standard textbook on the language. Recommended for use by Pascal/Z, Pascal/M and Pascal/MT users **$12**

☐ **THE C PROGRAMMING LANGUAGE**—By Kernighan and Ritchie. The standard textbook on the language. Recommended for use by BDS C, tiny C, and Whitesmiths C users . **$12**

☐ **STRUCTURED MICROPROCESSOR PROGRAMMING**—By the authors of SMAL/80. Covers structured programming, the 8080/8085 instruction set and the SMAL/80 language **$20**

☐ **ACCOUNTS PAYABLE & ACCOUNTS RECEIVABLE—CBASIC** book by Osborne/McGraw-Hill **$20**

☐ **GENERAL LEDGER—CBASIC** book by Osborne/McGraw-Hill **$20**

☐ **PAYROLL WITH COST ACCOUNTING—CBASIC** book by Osborne/McGraw-Hill .. **$20**

★ ★ ★ ★ ★ ★ ★ ★

* Program names trademarked

† Recommended system configuration consists of 48K CP/M, 2 full size disk drives, 24 x 80 CRT and 132 column printer.

Ⓜ Modified version available for use with CP/M as implemented on Heath and TRS-80 Model I computers.

Ⓛ User license agreement for this product must be signed and returned to Lifeboat Associates before shipment may be made.

Ⓘ This product Includes/eXcludes the language Ⓧ manual recommended in Condiments.

Ⓢ Serial number of CP/M system must be supplied with orders.

Ⓩ Requires Z80 CPU.

★ ★ ★ ★ ★ ★ ★ ★

Ordering Information

MEDIA FORMAT ORDERING CODES.
When ordering, please specify format code.

LIFEBOAT ASSOCIATES MEDIA FORMATS LIST. Diskette, cartridge disk and cartridge tape format codes to be specified when ordering software for listed computer or disk systems. All software products have specific requirements in terms of hardware or software support, such as MPU type, memory size, support operating system or language.

Computer system	Format Code
Altair 8800 Disk	See MITS 3200
Altos	A1*
Apple + SoftCard 13 Sector	RG
Apple + SoftCard 16 Sector	RR
AVL Eagle	RB
BASF System 7100	RD
Blackhawk Single Density	Q3
Blackhawk Micropolis Mod II	Q2
CDS Versatile 3B	Q1
CDS Versatile 4	Q2
COMPAL-80	Q2
Cromemco System 3	A1*
Cromemco Z2D	R6
CSSN BACKUP (tape)	T1#
Delta	A1*
Digi-Log Microterm II	RD
Digital Microsystems	A1*
Discus	See Morrow Discus
Durango F-85	RL
Dynabyte DB8/2	R1
Dynabyte DB8/4	A1*
Exidy Sorcerer + Lifeboat CP/M	Q2
Exidy Sorcerer + Exidy CP/M	Q4
Heath H8 + H17/H27	P4
Heath H89 + Lifeboat CP/M	P4
Heath H89 + Magnolia CP/M	P7
Helios II	See Processor Technology
Horizon	See North Star
iCOM 2411 Micro Floppy	R3
iCOM 3712	A1
iCOM 3812	A1*

Computer system	Format Code
iCOM 4511 5440 Cartridge CP/M 1.4	D1#
iCOM 4511 5440 Cartridge CP/M 2.2	D2#
IMS 5000	RA
IMS 8000	A1*
IMSAI VDP-40	R4**
IMSAI VDP-42	R4**
IMSAI VDP-44	R5**
IMSAI VDP-80	A1**
Intecolor	See ISC Intecolor
Intel MDS Single Density	A2
Intel MDS Double Density	A5
Intertec SuperBrain DOS 0.1	R7
Intertec SuperBrain DOS 0.5-2.X	RJ
Intertec SuperBrain DOS 3.X	RK
ISC Intecolor 8063/8360/8963	A1
Kontron PSI-80	RF
Meca 5¼"	P6
Micromation (Except TRS-80 below)	A1*
Micropolis Mod I	Q1
Micropolis Mod II	Q2
MITS 3200/3202	B1
Morrow Discus	A1*
Mostek	A1
MSD 5¼"	RC
North Star Single Density	P1
North Star Double/Quad	P2
Nylac Single Density	Q3
Nylac Micropolis Mod. II	Q2
Ohio Scientific C3	A3
Onyx C8001	T2#
Pertec PCC 2000	A1*
Processor Technology Helios II	B2
Quay 500	RO
Quay 520	RP
RAIR Single Density	R9

Computer system	Format Code
RAIR Double Density	RE
Research Machines 8"	A1
Research Machines 5¼"	RH
REX	Q3
Sanco 7000 5¼"	RQ
SD Systems 8"	A1*
SD Systems 5¼"	R3
Sorcerer	See Exidy Sorcerer
Spacebyte	A1
SuperBrain	See Intertec
Tarbell	A1*
TEI 5¼"	R3
TEI 8"	A1*
Thinkertoys	See Morrow Discus
TRS-80 Model I 5¼"	R2
TRS-80 Model I + FEC Freedom	RN
TRS-80 Model I + Micromation	A4*
TRS-80 Model I + Omikron 5¼"	RM
TRS-80 Model I + Omikron 8"	A1
TRS-80 Model I + Shuffleboard 8"	A1
TRS-80 Model II	A1*
VDP-40/42/44/80	See IMSAI
Vector Graphic	Q2
Vector MZ	Q2
Versatile	See CDS Versatile
Vista V80 5¼" Single Density	P5
Vista V200 5¼" Double Density	P6
Zenith Z89 + Lifeboat CP/M	P4
Zenith Z89 + Magnolia CP/M	P7

Prices reflect distribution on 8" single density diskettes. If a format is requested which requires additional diskettes, a surcharge of $8. per additional diskette will be added.

Prices F.O.B. New York. Shipping, handling and C.O.D. charges extra.

Manual cost applicable against price of subsequent software purchase.

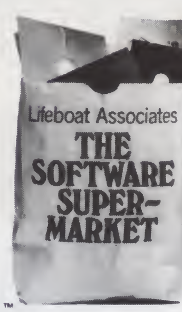
The sale of each proprietary software package conveys a license for use on one system only.

* Single-Side Single-Density disks are supplied for use with Double-Density and Double-Side 8" soft sector format systems

** IMSAI formats are single density with directory offset of zero.

\# A media surcharge of $25 for orders on tape formats T1 and T2 and of $100 for orders on disk formats D1 and D2 will be added.

The list of available formats is subject to change without notice. In case of uncertainty, call to confirm the format code for any particular equipment.



Lifeboat Associates, 1651 Third Avenue, N.Y., N.Y. 10028 **(212) 860-0300** ™The Software Supermarket is a trademark of Lifeboat Associates
International Telex: 220501, Domestic Telex: 640693

Put 16 bits on the S-100 bus with Marinchip Systems' M9900 CPU board.

Assemble a Super Business System

By Dr. Tom Lukers

I had no intentions of building an entire system from scratch.

I was already using a portable 6800-based system that included 40K RAM, triple minifloppies, an ADM-3A terminal and a quality printer, along with a very good text editor/processor package and a variety of custom software. It was, and still is, a good system.

But I needed an additional system for a variety of reasons.

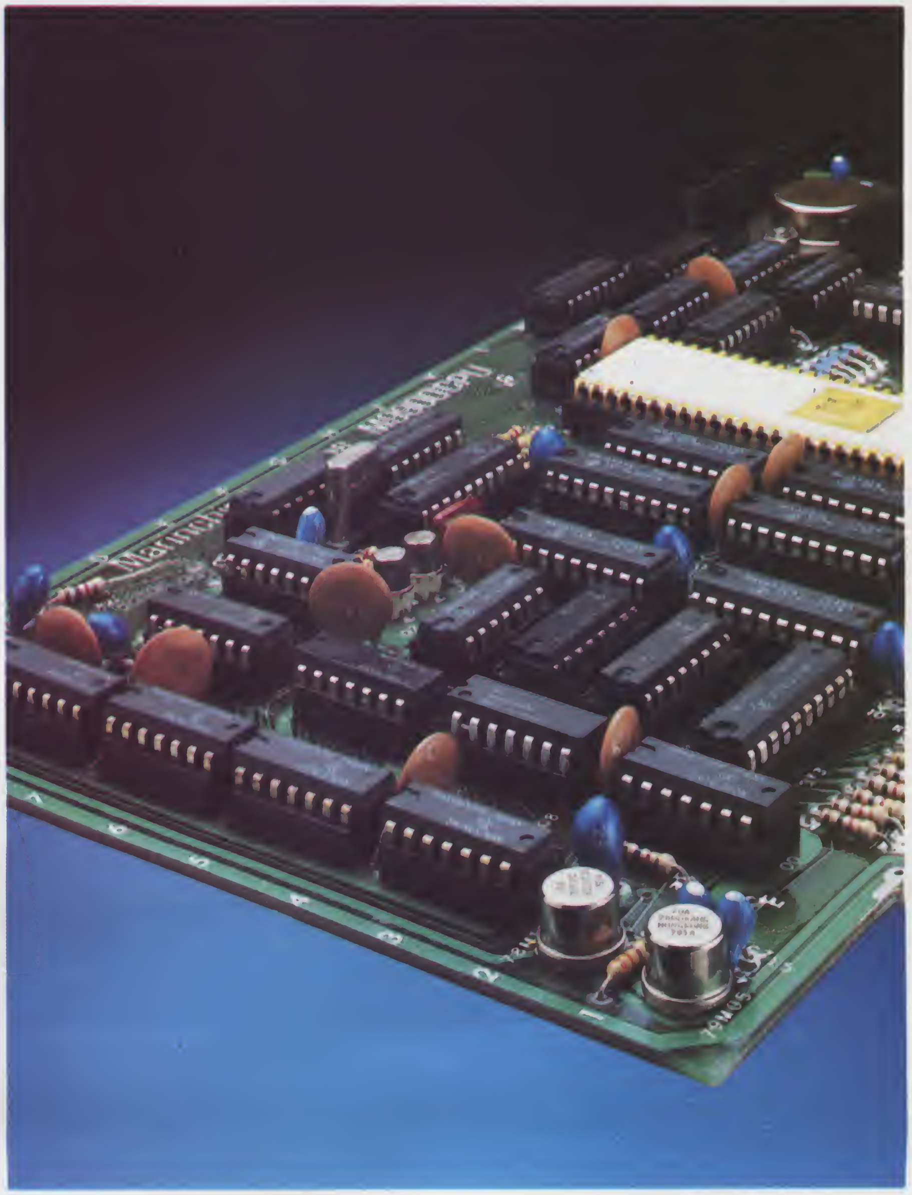
First, I wanted a powerful 16-bit computer with a good text editor/processor and a multitasking, multiuser operating system. I wanted a system that I could use with only one terminal and one printer, but that was easily expandable to run with several terminals, printers and other peripherals. I wanted to be sure that the software was already developed for driving a hard disk drive of at least ten megabytes.

And I wanted as standard system software the ability to use a modem to communicate with clients' mainframes, which requires some pretty good software.

These were not my only considerations. I also wanted to move into 16-bit technology with a chip family from a long-time computer manufacturer. Next, I hoped to investigate new technology as it came along. Finally, I wanted to be able to select peripherals and accessories from a wide range of manufacturers.

This article describes my solution—a 16-bit computer based on a Marinchip Systems M9900 CPU board run

Dr. Tom Lukers, 3625 Hendrick Drive, Plano, TX 75074.



Marinchip Systems' M9900 CPU Board. *(Photo by Martin Paul.)*

With the Heathkit H19 terminal on the left, the Diablo 1620 printer in the center and the TEI MCS-112 mainframe on the right, the installation is neat, compact and powerful. Dual Shugart SA800-1 disk drives are on the shelf underneath the mainframe and printer. Not shown is the Sunny International model R3 power supply on the shelf next to the drives. A cabinet and power supply for the drives would have cost about $200. Instead, I mounted the whole thing on a plywood sheet on one shelf of a stereo cabinet. The complete system is controlled by one power switch located under the near end of the bottom shelf.

on the S-100 bus.

The CPU board uses the Texas Instruments TMS9900 microprocessor and operates under a system disk executive with 64K RAM and dual eight-inch floppies. It is expandable to run under a multiuser, multitask operating system, with I/O peripherals and disk storage limited only by cost.

By building almost all of the system from kits, I saved about $1000 over what the units would have cost off the shelf. And, through a stroke of luck, I saved about another $1000 on the dual eight-inch floppy disk drives and power supply, plus several hundred dollars more on a high-quality daisy wheel printer.

This system was built from scratch, but if you already have an eight-bit S-100 system, you can upgrade to 16 bits just by replacing two of your present boards. And you can keep the same memory boards and peripherals that you're using now. Although this system has 64K of dynamic RAM, organized as 32K of 16-bit words, the 9900 CPU board is designed to work with either eight-bit or 16-bit memory boards, or a mix of both.

Even if you're not using an S-100 system, you still may be able to use your same video terminal, disk drives and printer.

The System

The system is made up of three groups of equipment, plus system software. The CPU group comprises the CPU, PROM/RAM/SIO and memory boards, plus the S-100 mainframe/motherboard. Heathkit's model H19 terminal, the Diablo 1620 printer and the SSM I04 I/O board make up the I/O group, while two Shugart 800-1 drives, a Tarbell 1011 disk controller and Sunny International Power Supply R3 make up the disk drive group.

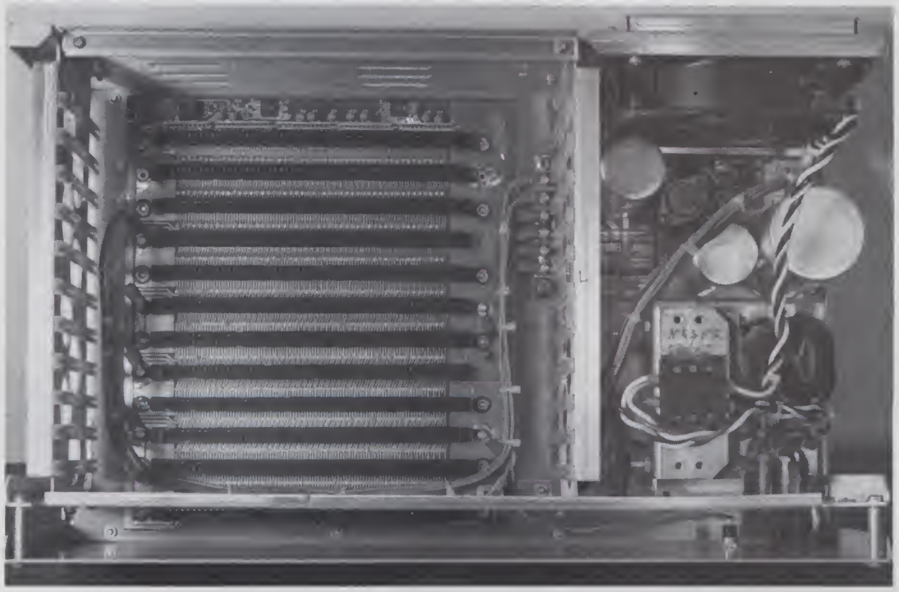
If you already have any or all of these units in a working system, just skim that part and move on. If you already have an S-100 computer system up and running, then you may be interested in how to upgrade to the M9900 16-bit system.

The Marinchip Concept

Several years ago the folks at Marinchip Systems (16 St. Jude Rd., Mill Valley, CA 94941) set out to design a 16-bit computer system to run on the S-100 bus, a bus that was designed largely for the Intel 8080 eight-bit microprocessor. The S-100 bus was not intended to be a bus standard. It was just the first—and for a long time, the only—bus around.

Marinchip Systems found that to design a 16-bit system that is compatible with the S-100 bus, they had to make the 16-bit system simulate some of the signals produced by the 8080. And this had to be done without sacrificing any of the power or performance of the 16-bit CPU.

One interesting problem, for example, was how to make the system mostly independent of the organization of the memory boards. If I already have an eight-bit system on the S-100 bus that works fine, do I really want to throw away $1000 that I have invested in 64K of memory boards



Top view of the TEI mainframe shows the clean, rugged design. The well-filtered, regulated power supply is to the right of the divider panel. The large wires connect to the constant voltage transformer at front and cooling fan in rear. Power on-off and reset are the only front panel controls. The unit is compatible with the proposed IEEE S-100 standard.

that are organized in eight-bit words? The designers at Marinchip considered this and other questions of compatibility.

The Marinchip 9900 CPU doesn't care whether the memory area is populated with eight-bit memory boards, 16-bit boards or a mixture of the two.

During a memory access, the SXTN-signal (pin 60) lets the CPU know whether the memory board is eight- or 16-bit. If the addressed memory board pulls this line low, the CPU knows that it is a 16-bit board.

And, as they point out in the M9900 manual, since this signal originates with the memory board itself, it is possible to mix the two kinds of boards.

As evidence of their success, the system operates just fine with the Tarbell Model 1011 disk controller, which was designed originally to operate in S-100 systems being driven by 8080 or Z-80 microprocessors.

This makes it nice for the paying customer. If I have trouble with my system, I can take any part of it to a local shop, where they can swap around components in other S-100 systems and quickly find the trouble.

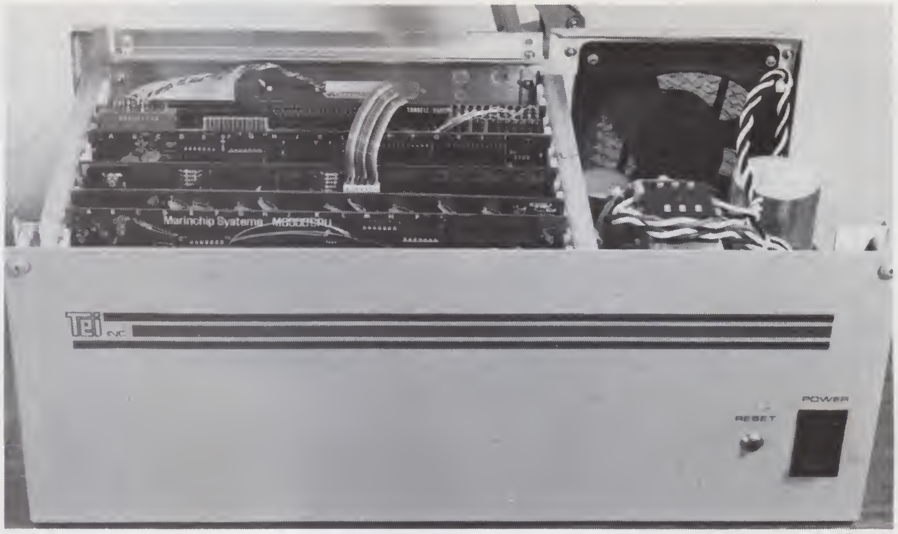
The CPU group is made up of the S-100 mainframe/motherboard and three printed circuit boards by Marinchip Systems. The 12-slot mainframe is by TEI, Inc., Houston, TX. The M9900 CPU board, the PROM/RAM/SIO board and 64K dynamic RAM board are by Marinchip Systems.

TEI Model MCS-112

One of my main criteria was a high-quality rugged mainframe with a heavy-duty power supply. The TEI model MCS-112 fills the bill.

The model MCS-112 features a completely assembled and tested 12-slot motherboard in a sturdy industrial-quality chassis and enclosure. The back panel is punched for eight DB25 and two DB37 connectors. The DB25s can be used for RS-232 connectors and the two DB37s for ribbon cables and clamps, such as the ribbon connector for the Tarbell disk controller. The front panel has a lighted power switch and a push-button reset switch.

The MCS-112 power supply has a constant voltage transformer. The supply is rated at 8 V @17 amperes, plus and minus 16 V @2 amps, over an input voltage range of 95 to 130 V ac.



Just behind the front panel of the TEI S-100 mainframe is the Marinchip Systems M9900 CPU board, with the 64K dynamic RAM next. The right-hand ribbon connector exits from the SSM IO4 board. Just behind the SSM IO4 board is the Marinchip PROM/RAM/SIO board, with a 32-pin edge connector on J2 providing interface with the system terminal. At the very rear is the Tarbell disk controller with its large ribbon cable exiting at a slot on the left of the rear panel. The right-hand section of the mainframe contains the heavy-duty power supply, constant voltage transformer and cooling fan.

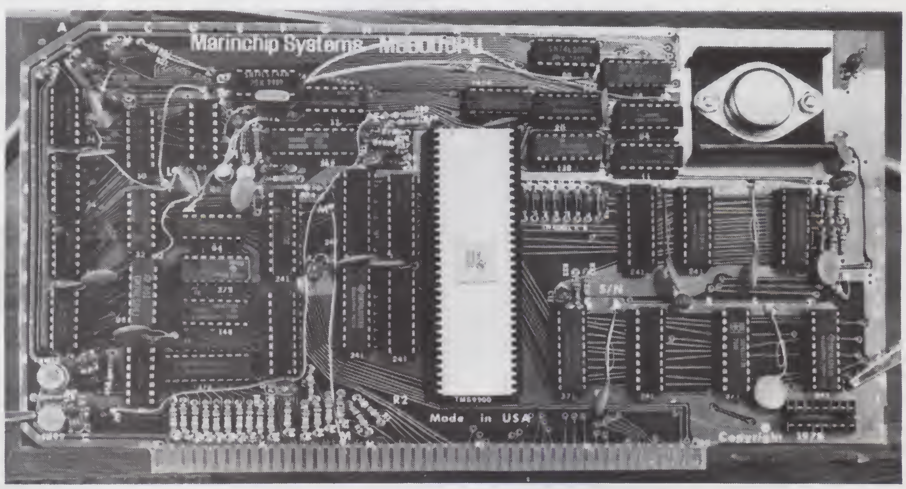
A built-in blower motor provides forced-air cooling. The blower motor is a possible source of complaint. It is noisy, although no noisier than any other fan/blower motor on equipment of this size. If the noise of a blower motor bothers you, then you should consider either locating the mainframe out of hearing or possibly choosing another mainframe. As for myself, I prefer the noise to the problems frequently encountered from poorly cooled equipment.

The model MCS-112 comes with a no-frills, but adequate, 14-page manual. The manual includes a schematic, parts layout drawings, a parts list, a brief theory description and instructions for test and checkout. The unit is described as being compatible with the proposed IEEE S-100 bus standard.

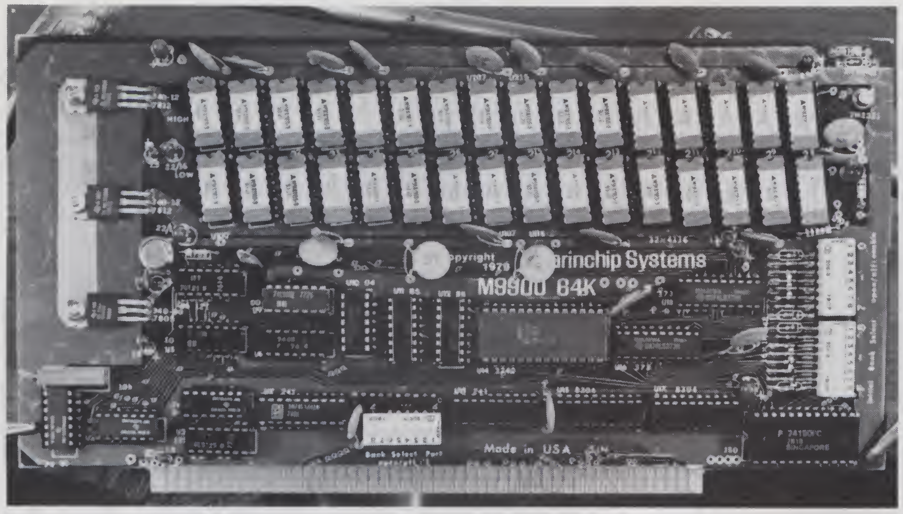
Marinchip Systems M-9900 CPU

When Marinchip Systems designed the 9900 CPU board, they produced a high-quality product and made it available either in kit form or assembled and tested. I bought the kit.

The board has plated-through



The Marinchip Systems M9900 CPU board shows the relative size of standard 7400 series TTL devices and the monster 64-pin TMS9900 microprocessor. The board comes either as a kit or fully assembled and tested. Kit construction is facilitated by the silk-screened board and clear, easy to follow instructions. For hardware types, Marinchip Systems also provides a comprehensive theory section explaining how they were able to make a 16-bit computer work on a bus designed for eight-bit systems.



The 64K dynamic RAM board features a clean, elegant design with 32 4116 dynamic RAM chips, controller and bank select/address decode DIP switches. The memory is organized as 32K of 16-bit words and operates from two to three times faster than comparable eight-bit boards.

holes, coating to minimize the possibility of solder bridge during construction and a grid layout to help in correctly locating the parts on the component side of the board. In addition, each IC socket location is silkscreened to show the type of IC chip that goes in the socket.

Instructions for building the board are good. Documentation for the 9900 CPU board includes 20 pages of step-by-step assembly instructions, a 12-page section describing theory of operation and an eight-page compatibility bulletin describing a number of system components manufactured by various companies. The bulletin gives information on how well the components work with the Marinchip 9900 CPU, and vice versa. It is a useful guide when selecting system components.

Construction is straightforward, with only a few jumper connections required at the end of construction. Instructions for placing the jumpers are concise and easy to follow.

If you have experience in placing ICs in their sockets, you will probably have no trouble installing the 64-pin 9900 microprocessor. Otherwise, carefully follow the direction provided by Marinchip Systems and you should be OK.

All sockets are provided for the 36 ICs, including the one for the 9900 CPU. In addition to the ICs, the other parts include three on-board three-pin voltage regulators, 39 resistors, 33 capacitors, one inductor and one crystal, plus all the small hardware necessary for mounting the regulators.

I'm a careful, methodical worker, and it took me about eight hours to build and check out the board, following the tests outlined in Marinchip's instructions.

Though I work under a magnifying lamp, I still failed to solder one of the IC socket pins. The board passed all of the voltage and waveform tests but still failed some of the final tests. It is extremely difficult to troubleshoot a system when every item in it is new and built mostly from kits. After trying to find the trouble myself, I sent the board to Marinchip Systems. They gave me one-day turnaround, sending it back by UPS. It has been a long time since anyone gave me that kind of service at any price, especially after the sale.

Marinchip Systems PROM/RAM/SIO Board

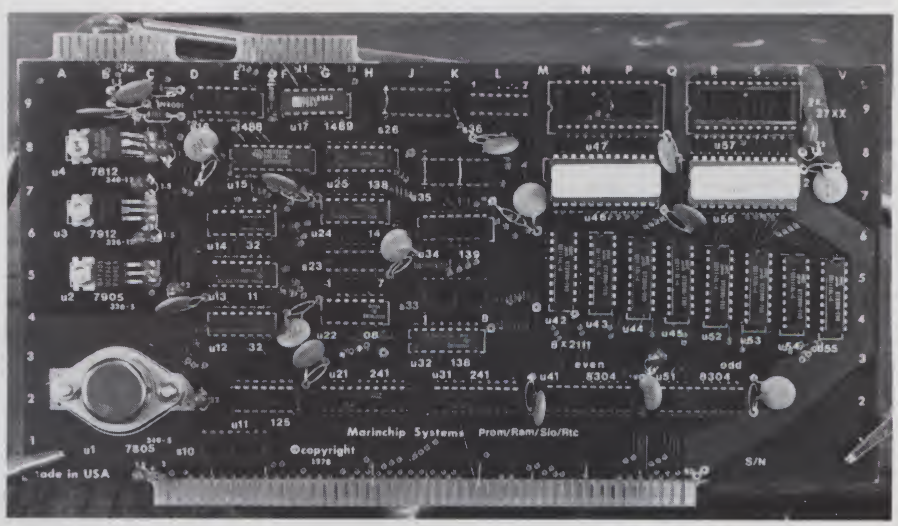
Typical of Marinchip quality, the PROM/RAM/SIO board provides all of the functions of a serial I/O, 2K PROM-based system monitor, 1K of scratchpad RAM and real-time clock, all on one coated board with plated-through holes and silk-screened layout. In addition to the system monitor PROM, the board provides for up to 8K of user PROM storage.

Documentation for the PROM/RAM/SIO board is thorough, with 16 pages of step-by-step construction notes, plus a 22-page booklet covering theory of operation and system configuration. Included in the guide are 11 pages of waveforms and schematics.

Also included are the instructions for determining what kind of jumper options you need with your particular system configuration, plus the instructions.

One of the unique characteristics of the Texas Instruments 9900 microprocessor is its communications register unit. The CRU makes it possible to address any bit out of a 4K bit address space at any time. Thus, you can input or output serially from one to 16 bits at a time.

The addressing and decoding techniques used in CRU input/output are different than those used in memory-mapped I/O. But there are distinct advantages, such as the size of the spe-



The PROM/RAM/SIO board provides serial I/O between the CPU and the terminal, plus a ROM-based debug monitor, real-time clock and scratch pad RAM. A version is also available to be used strictly as an I/O port. The two 32-pin edge connector ports at upper left are for I/O. The one on the left, J2, is for interfacing with the terminal. Only one of the connectors can be used at a time. The one on the right, J1, is for modem hook-up.

cial I/O chips used with CRU I/O. The TMS9902, for example, is much smaller than its counterpart in a memory-mapped I/O. If you compare the size of a 6850 ACIA, used in a Motorola 6800 system, you'll find that the 6850 is a 24-pin device, compared with the 18-pin DIP packaging of the 9902.

The 9900 microprocessor can be used in either or both types of I/O. In my system, I use the CRU I/O of the PROM/RAM/SIO board for the Heathkit H19 terminal, and I use memory-mapped I/O for the Diablo 1620 printer. Later, I will also be using memory-mapped I/O for modem communications.

64K Dynamic RAM Board

Marinchip Systems took an array of 32 type 4116 dynamic RAM chips and came up with a 64K dynamic RAM board organized as 32K of 16-bit words. They use a bank-select memory scheme to implement its multiuser networking operating system. The 64K dynamic RAM board is designed to support the bank-select feature. Since the board is designed for 16-bit transfers, it operates about three times faster than a comparable eight-bit board. But this 16-bit transfer feature also means that this board cannot be used in another system operating with eight-bit data transfers.

Marinchip also states that this board is not compatible with DMA data transfers. Devices using DMA data transfers to and from eight-bit memory boards in the system have no interaction with the Marinchip 64K dynamic RAM board. But the board cannot handle DMA transfers to and from itself. The board comes assembled and tested, with full instructions for setting the DIP switches for memory-map assignment.

Disk Drive Group

The disk drive subsystem consists of two Shugart SA800-1 floppy disk drives, a Tarbell Model 1011 floppy disk controller and a Sunny International Power Supply, Model R3.

Current Marinchip Systems software supports single density operation only, although a double density software system is nearing completion. For single density operation, the Tarbell model 1011 controller is a good compromise between cost and quality. The disk subsystem produces soft-sectored disks that are compatible with the IBM 3740 format for single density recording. This means that there are 77 tracks, numbered 0 to 76, with 26 sectors on each track, numbered 1 to 26. Each sector contains 128 bytes of information. This gives a total storage capacity of about 256K bytes per disk, if you're using single-sided disks.

Shugart's Floppy Disk Drives

If you are lucky enough to find a couple of the Shugart Model SA800-1 floppy disk drives, you will be pleasantly surprised to find that they are low in cost and high in performance. Note that this is the model SA800-1 and not the model 801. You probably won't find the model SA800-1 new. This version of the model SA800 was OEM-built by Shugart for a large company to be used in word-processing systems. There seems to be quite a few of them around on the used equipment market. I was fortunate enough to find two for less than $500, including the drive electronics, but without either a case or power supply.

The documentation with the drives included an instruction manual, complete with theory of operation, waveform diagrams, adjustment procedures and schematic diagrams. All of this information is for the Model SA800/801. The manual clearly shows the major differences between a standard SA800 and the 801. But the only information concerning the model SA800-1 is a statement that the 800-1 contains its own data separator. The schematics make no references to the model 800-1.

I tried to get the drives to work and achieved some success. But operation was erratic. Sometimes everything would work fine; at other times, it would bomb.

I visited the local Shugart sales office, where the sales engineer gave me a set of the correct schematics and the parts layout drawing and parts list.

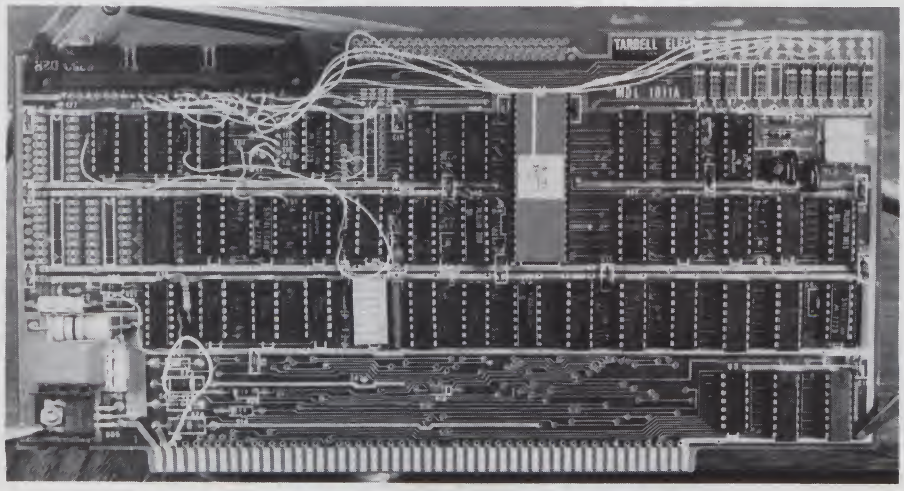
I made all the strapping arrangements exactly as shown on the new schematic, and most of the problems seemed to disappear. One trouble persisted, however. Although the drives seemed to work fine in all other respects, I could not always copy a full disk from one drive to the other. When they did work, they seemed a little slow.

I took it to a technician, who added a lot of jumpers to the drive electronics. He said that, in effect, he had converted the drives to 801s. So, the drives do everything they're supposed to do, and they do it faster and quieter than they ever did before. And here's the final payoff: total cost for the repair work was $60.

Incidentally, I asked the Shugart sales engineer whether the 800-1 drive could be used in double density systems. He said that the 800-1 should work fine in double density systems without any problems at all.

Tarbell 1011 Disk Controller Board

The Marinchip M9900 system was designed to operate with standard



The Tarbell model 1011 disk controller can be used with a large number of different disk drives, both eight-inch and minifloppy. I/O to the drives is by the ribbon connector plug at the upper left. The DIP switch at lower center is set for use in the M9900 system, with all switches off. This board has an on-board bootstrap ROM, which is not usable in the Marinchip M9900 system, since the ROM contents are in 8080 machine language. It is useful on the troubleshooting bench, though, when servicing the drives.

Model R3 dual drive power supply by Sunny International provides all dc voltages necessary for the two eight-inch drives. You have to provide your own interconnecting harness and connectors and use care when soldering to the power supply terminals. But at about $65, plus shipping, it is still one of the best buys that you'll find advertised in Kilobaud Microcomputing.

S-100 peripherals. As an example, the Tarbell model 1011 disk controller board has been around long enough that it is one of the "tried and true" industry standards.

If you're new to the S-100 bus (as I am) and ask a few questions of people who've been using it for a while, you'll get a lot of good comments about the Tarbell controller.

The Marinchip compatibility bulletin lists it as one of the preferred controllers for the M9900 system. A couple of others are recommended, but the Tarbell controller seems to be the best overall compromise between quality and cost. And if you do use the Tarbell controller, you won't be forced to use any one particular kind of disk drive.

You can buy it either as a kit or assembled and tested. Kit construction is straightforward, with no hidden traps. The built-in strapping options make it useful with a wide variety of disk drives. Strapping information is given for most of the more popular disk drives, both eight-inch and mini.

If you use the Tarbell controller with the Shugart eight-inch floppies, follow the strapping information in the Tarbell manual for the SA800/801. A question does arise concerning jumpers 146, 147 and 148. In my system 146 is jumpered to 148.

Sunny International Floppy Disk Power Supply

Adequate dc power for the SA800-1 disk drives is provided by the model R3 power supply by Sunny International. Voltages of +5 V @8 A, −5 V @1 A, and +24 V @5 A are provided in an open-frame construction with good fuse protection and adequate heat sinking. The three dc supplies use three-terminal regulators, with outboard series-pass transistors used in the +5 V and +24 V supplies. As shipped, the power supply has a two-wire ac line cord, which I replaced with a three-wire cord. I connected the third (green) wire to the chassis.

You have to build your own connecting harness to interface the power supply with the disk drives. They recommend 14-gauge wire between the power supply and drives, and 12-gauge for common ground connection. The drives require a mating connector, AMP part number 1-480270-0, with AMP pins, part number 60619-1, or equivalent. The connector for ac power to the drive motors is AMP P/N 1-480305-0, with AMP pins P/N 60620-1, or equivalent. If all else fails and you can't get the connectors locally, try KA Electronic Sales in Dallas. They were able to get the connectors for me.

The Diablo 1620 printer-I/O terminal has a numeric keypad, tractor feed mechanism and print speeds from ten to 45 characters per second. Speeds are switch-selectable from a front panel mounted just under the pop-off nameplate. The top right row of keys are for reset, form feed, set top of form, scroll and power on-off. The tractor feed mechanism pops off in seconds.

The power supply is built on a printed circuit board, which is mounted to an aluminum chassis with standoffs. Connection to the circuit board is provided by terminals, which are soldered to the foil on the bottom of the board.

The power supply is built on a printed circuit board, which is mounted to an aluminum chassis with standoffs. Connection to the circuit board is provided by terminals, which are soldered to the foil on the bottom of the board.

A problem arises in connecting such large wire to the pins. Sufficient heat to make a good electrical connection is also enough to melt the solder that connects the pin to the foil. Use care that this doesn't happen. It would help if larger connector pins were provided.

Heath H19

I found the Heath Model H19 to be pretty much as reviewed in *Kilobaud Microcomputing* ("Heath's H19: A Detailed Look at a Super Terminal," February 1980, p. 58). Construction took me about 27 hours, not counting debugging. I did make a common construction mistake that cost some time during final checkout. I inadvertently interchanged the two large TO-3 power transistors located in the power supply. So, when I gave it the smoke test, it passed, giving lots of smoke. Two transistors, one potentiometer and a couple of diodes later, everything was fine, and has been since.

Diablo Daisy Wheel Printer

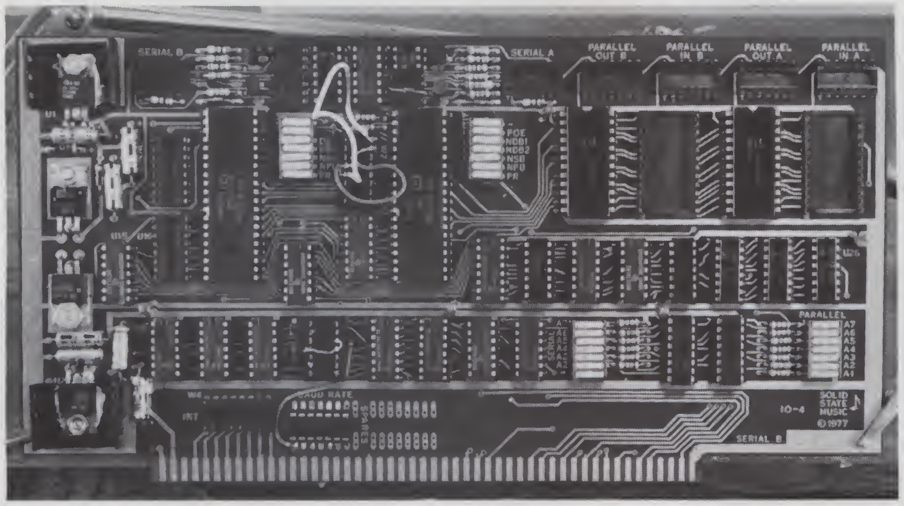
With print speeds from 10 cps to 45 cps at baud rates of 110 to 1200 baud, the microprocessor-controlled model 1620 HyType II printer is probably one of the finest printers on the market. It is actually an I/O terminal as well as a printer. And a wide variety of type styles adds to its versatility.

Except for 1200 baud, the baud rates are switch selectable. For 1200 baud operation, a jumper is required on one of the circuit boards. When the 1200 baud jumper is in place, the speed switch has no effect.

The model 1620 will operate at baud rates to 300 baud with no handshaking required. For 1200 baud, however, you'll need an ENQ/ACK protocol in the software driver.

The HyType II printer mechanism is used in a wide variety of printer mechanisms; the model 1620 is only one of a large line. Some versions use metallized print wheels. Mine uses the plastic wheels. Some users think that the metallized print wheels give better printing quality. If you decide to buy one of these beauties, you should know three things.

First, if you want the metallized print wheel, you must specify before you buy. Printers designed for the plastic print wheels won't work with the metal ones. Second, repairs were quite expensive. Although I have had nearly a year of good service under



SSM Model IO4 I/O board provides two serial and two parallel I/O ports. The two DIP switches at lower right do the address decoding for the serial (left) and parallel (right) ports, respectively. The upper two DIP switches set the mode parameters for the two serial ports. Above and to the right of the top right mode switch is a 16-pin IC socket which accommodates a 16-pin header to a ribbon cable for I/O to the printer.

heavy-duty conditions, I did have a servo board fail. Third, I have seen the model 1620 advertised for over $3000 new, but you might be able to find one for less than $2500 used.

Documentation for the Model 1620 includes a fine operator's manual and, for $20, a superb maintenance manual. The maintenance manual contains complete schematic and logic diagrams, the theory of operation and a parts list. It is well worth the price.

SSM Model 104 2P + 2S I/O Board

The SSM Model 104 I/O board is a high-quality kit that provides two serial and two parallel I/O ports for S-100 microcomputer systems. The kit uses a coated board with plated-through holes. The coating minimizes the possibility of solder bridges during construction, and the plated-through holes ensure that the etch makes good contact on both sides.

Assembly instructions are good. SSM provides information for configuring the board for a variety of systems. They also give information on a hardware interface with the Heathkit H14 printer's busy signal.

The Heathkit manual gives instructions for modifying the printer to produce the complement of the busy signal. If you have made this modification, then you will need to modify the circuit shown in the SSM manual. All you have to do is leave the NAND gate being used as an inverter immediately following the zener diode added during the SSM modification.

A number of strapping options must be implemented to make the SSM board work in the 9900 system. Some of them involve dual inline (DIP) switches; one involves soldering a wire on three of the pins of a 16-pin DIP header. The switch settings are for setting the port addresses and the baud rate, parity, and so forth, on the serial interface. Set these according to your system configuration.

The baud rate between the computer and printer must agree. So, decide what baud rate you're going to use with your printer. Then run a strap from W2 pins 11 and 12 to the pin that corresponds to the baud rate you're using. For example, if you're using 300 baud, use pin 4. But for 1200 baud, use pin 2. And for 4800 baud, use pin 6.

Using the SSM Board with the Diablo 1620 Printer

If you're using the SSM board with the Diablo model 1620 printer, you can run the printer at 300 baud without handshaking of any kind, either hardware or software. But if you run the printer at 1200 baud, you will have to strap it inside for 1200 baud rate. And you will have to provide a software handshake in your printer I/O routine. I am running mine at 300 baud until I get the time to write the handshaking routines.

Without handshaking, the printer occasionally loses a character. I ran it for nearly a year on another system at 1200 baud with handshaking; it hardly ever lost a character. So, the handshaking is important.

System Software

Standard system software includes a disk executive, a BASIC interpreter, a ROM-based debug monitor, an excellent text editor and a good text processor.

The BASIC is similar to Microsoft 12K BASIC, but does not include double precision, random access files or PRINT USING. However, an extended commercial BASIC is available at extra cost; it has all these features, plus 16-digit accuracy and CHAIN with common variables. It also lets you save either the source or compiled code. Both types of BASIC include long variable names, integer arithmetic, sequential files and the full complement of string and mathematical functions.

The executive supports named files on disk, contains all I/O drivers and handles peripherals the same as any other files. The executive allocates system memory to programs automatically, so that programs can automatically adapt to the amount of free memory.

Other built-in features include an editor for editing from the local terminal, an assembler and a linker. All system calls are compatible with the Marinchip Network Operating System, so that any programs developed to run under the disk executive will also run under the NOS if you later add it to the system.

If the system encounters an unknown command, it searches the user files for a program by that name. If one exists, it loads and executes it. This lets you customize your system without having to modify the executive itself.

I like the set of disk executive troubleshooting routines called Brainstorm. They are helpful in troubleshooting system memory and the main processor board.

Other built-in features of the executive include a routine for programming EPROMs on the Cromemco Bytesaver PROM Programmer, plus other disk-handling functions, such as disk copy, name, rename, initialize disk and file diagnostics.

One feature, which is really part of the Debug monitor, but runs under direction of the system executive, is the T utility for emulating a terminal when communicating with other systems.

I would like to see one other feature incorporated into future versions of the executive—the ability to

type ahead. I hate waiting for one command to be executed before I can type in the next one. I would like to see this feature linked with the editor, too.

Text editing is easy using the Marinchip text editor. It is a line-oriented context editor with two modes, input and edit. This makes it easy to create new files or to edit and modify existing ones.

For example, making changes within a line is easy. Eleven commands are available under the general command ALTER just for making changes within a line. When in the edit mode, any of these commands can be used at any time within a line. This makes the editor flexible for within-the-line changes. Other commands are available for general editing and formatting.

One unique feature of the editor is its automatic paging to disk files when the size of a file approaches being too large for available memory. This paging to the editor-backing files is automatic, without any user intervention.

I'd like to see one feature added to the editor. The editor has provision for line numbers, and I'd like to be able to toggle the line numbers on or off. As it is, if you want a line displayed with its number, you must command it each time.

I would also like the ability to read in a specified number of lines from a disk file. As it is, if you want to read any part of a file from disk, you must read in the entire file. When editing large files, it helps to be able to read in 100 or 200 lines, edit and spool it out to a new file, repeating this process until the entire file is done. This is valuable when a file has 1500 or 2000 lines of text.

Even without these features, though, I like the Marinchip text editor.

The Marinchip Systems text processor has a number of features that make it a powerful, flexible and useful tool in writing. The surprising thing is that it comes as standard Marinchip software, and not as an option.

The text processor makes it possible to automatically format text, set up footers and headers, set up justified right and left margins and assign page numbers. It has a decimal-numbering mode for paragraphs, so that it will automatically number the paragraphs and generate a table of contents for the document. It can produce multiple-column output so that text can be printed in two or more columns.

Macro capability makes it possible to predefine text—or even entire documents—to be called up as part of the document being processed. During form letter generation, for example, text from more than one file can be automatically called up and included in the text of the letter.

The processor commands are embedded in the text during editing and then acted upon during processing.

The Marinchip text processor probably does just about anything that you'd need in a text processor for general business correspondence.

Room for Improvement

I'd like to see a few enhancements added to the text processor. First, I would like to be able to relate the position of the page number to the page number itself. That is, I would like to be able to set the processor so that on even-numbered pages the page num-

The Marinchip text processor probably does just about anything that you'd need in a text processor for general business correspondence.

ber appears on the left side of the page and on odd pages appears on the right side. Also, I'd like to be able to set up some test for the execution of conditional macros and set traps for the execution of macros or other processor commands. And I'd like access to some number registers that I could define as needed for setting up my own formatting profiles. Finally, I'd like the ability to have the processor output selected pages of processed text.

For example, if I have a 15-page file and have just revised pages 12 through 15, I'd like to run the file through the processor and output only those pages. As it now stands, I have to output the entire file just to see how the last four pages look.

Another problem is a glitch that occurs on page boundaries during processing. For some reason, the printer hesitates and sometimes misses a character. I talked with Marinchip, and they were already aware of the problem. A revised copy of the editor/processor package solved the problem.

One of Marinchip's policies that I like concerns software cost. They want to keep it as painless as possible. So, as they revise their software, they make the revised versions available on eight-inch disks at $6.30 per disk. This barely covers the cost of the disk.

Extra-cost software for the system includes the extended commercial BASIC, the network operating system, a seven-pass Pascal compiler and a version of BASIC called QBASIC.

The network operating system, or NOS, makes it possible to have multiple terminals, multiple printers and other I/O devices, all seemingly operating at the same time. To use the NOS, you have to devote a 64K board of memory to the CPU and one additional 64K board for each terminal. You can have one terminal operating with the NOS with only one 64K board, but in that case the CPU gets 32K and the terminal is limited to only 32K for its use. The best arrangement is to reserve 64K for the CPU and 64K for each terminal in the system. In anticipation of small businesses using the system under NOS, Marinchip sells a 40 megabyte hard disk drive system and a 165 cps line printer.

Their QBASIC is described as being so close to CBASIC that software written to run under CBASIC can be run under QBASIC with only minor changes. They have implemented the Osborne general ledger, AR/AP and payroll packages under QBASIC.

For folks who may already have invested a bundle in software written in CBASIC running under CP/M, Marinchip has developed a disk utility to let you load CP/M-compatible ASCII files into the Marinchip M9900 system.

The seven-pass Pascal compiler is a sequential Pascal, with great differences between it and other Pascals. They intended their Pascal to be used more for software and applications development than for other purposes.

There is just one thing that I'd like to see added to the system software: a good driver that implements all of the bells and whistles on the Diablo model 1620 printer. This would include backspace/underscore, superscripts and subscripts, plus a number of routines for using the graphics capabilities of the printer.

Putting It All Together

Once all the units have been built, you can start putting it all together into a working system. Four things need to be checked: the interfaces between the CPU and terminal, the CPU and printer, the disk drive to CPU interface and the address switches in the 64K dynamic RAM board if you are using it.

On the PROM/RAM/SIO board, you will use the upper left 32-pin edge connector, J2, for interfacing the CPU with the terminal. There is a one-to-one correspondence between these pins and the pins on the DB25 terminal connector. I used pins 2, 3, 4, 5 and 7. At the DB25 connector, I shorted pins 6 and 8.

Setting the addresses for the 64K RAM board and the SSM I/O board will depend on the instructions you receive from Marinchip Systems. This, in turn, depends on your particular system configuration.

If you are using the Tarbell controller, you will set all sections of the DIP switch to off for normal running. During the final test, you will set section 6 on to prevent accidental erasure of your disks.

The Tarbell controller contains a ROM-based bootstrap program that was designed for 8080/Z-80 S-100 systems. Since it was written in 8080 machine language, you won't be able to use it in the 9900 system. It will pay off, however, if you need to have the disk drive subsystem repaired or adjusted.

You must debug your system after construction is complete. I allowed ten days for debugging this system. During this time, I had two solder bridges, three bad IC chips and one IC pin that was unsoldered, plus the goof with the Heath H19 terminal.

Since all of the kits were of such high quality, and because of the usual high reliability of IC components, I feel that I may have inadvertently damaged the chips myself. I worked during cold weather, when LSI chips are most susceptible to damage from static electricity.

In any event, if you have trouble and can't correct it yourself after a reasonable effort, then call Marinchip Systems. They will bend over backwards to help. And if you do send the boards to them, they can find the bad board in a matter of minutes and then find the trouble in another few minutes.

If you want to move up to 16-bit technology, if you want a business system that can handle two to four eight-inch floppy disk drives with expansion capabilities to over 40 megabytes of hard disk with multi-user capacity, then the Marinchip Systems M9900 could be the one for you. ■

A machine-language program that tightens up North Star BASIC programs.

Blank Removal

By Rinaldo F. Prisco

There are many reasons why you might want to remove spaces from a program. Such a procedure will conserve disk space and RAM, and increase execution speed.

In most cases, you won't gain much. Keeping the spaces will make programs more readable, and thus more easily maintained. But sometimes removing them might be advisable, or even necessary—you can write longer, faster programs with more variables in the same amount of space.

This 87-byte machine-language program will remove all unnecessary spaces occurring in a North Star BASIC program. It is in 8080/Z-80 code, and is thus much faster than similar programs written in BASIC. It is also easy to use; you simply enter the DOS, type GO REMOVE <CR> and you are done. The program even returns you to BASIC.

Program Design

The idea behind the program is simple. You use two pointers: the source pointer (DE register pair) and the target pointer (HL register pair). You examine each byte of the source code. If it is not the ASCII code for a space, it is written back to the target. The pointers start out even. As more spaces are removed, the HL pointer falls further behind the DE pointer.

You want to avoid removing spaces that occur in REM statements or between quotation marks so you check for them, jumping to the proper routine when they are encountered.

You also want to avoid removing "false" spaces. The ASCII code for a space is 32 (20H). This byte can occur in line numbers or in line numbers referenced by GOSUB- or GOTO-type statements. For example, "GOSUB 8200" will contain the byte 32 since 8200=32*256+8. You must recognize line numbers and their references to avoid being fooled.

Line numbers themselves are fairly straightforward since they always occupy the second and third bytes of a new line (the first byte contains the number of bytes in the line). New lines begin with the byte that follows a "true" ASCII CR, the byte 13 (0DH). Whenever a CR (not in a line number or reference) is encountered, you update the number of characters in the line being revised and begin scanning a new line.

Line numbers referenced by GOSUB- or GOTO-type statements are always prefaced by the byte 154 (9AH) and occupy the next two bytes (low byte, high byte).

Program Description

The program in Listing 1 is designed for the standard configuration of Release 4 North Star BASIC. It is easily modified for other versions. The only changes that might be required are in the address where the first program byte appears (5B93H in Release 4) and the address where BASIC is entered, preserving loaded programs (2A04H in Release 4).

The program was assembled at 0 but could be assembled to reside anywhere other than RAM occupied by the DOS, the BASIC interpreter or the BASIC program. It can even be assembled to reside in high BASIC RAM. If so, you need not modify the BASIC RAM with MEMSET, since the code will be overwritten when the BASIC program is run.

The program consists of six routines: NEW, SKIP, HIT, QUOTE, REM and UDATE.

The NEW routine is used whenever a new line is begun. It first checks the length of the new line. If it is one, then the end of the program being processed has been reached and a jump is made to BASIC. Otherwise, the address of this byte in the target program is pushed on the stack and its contents put in the B register.

The SKIP routine is branched to whenever a line number or reference is encountered. In such cases the program skips over the next two bytes being processed (they are line number codes).

The HIT routine examines successive bytes, checking to see if they are line-reference flags (9AH), end of lines (0DH), the beginning of REM statements (8FH) or quotation marks (22H). If any such bytes are encountered, there is a branch to the proper routine. Otherwise, the program checks for "true" spaces (20H). If the byte is not a space, the program loops back to HIT for another byte. If it is, you decrease the target pointer (so the next nonspace byte will occupy the byte it addresses) and decrease the contents of the B register that contains the number of bytes in the target line.

The QUOTE routine just cycles the bytes until another quotation mark is encountered. The REM routine does the same thing until the end of a line. The UDATE routine updates the number of bytes in the revised target

Rinaldo Prisco is an associate professor of mathematics at the State University of New York at Oswego, NY 13126.

```
0000                  *                 REMOVE
0000                  *
0000                  *        REMOVES BLANKS FROM
0000                  *        PROGRAMS IN N* BASIC
0000                  *
0000                  *          RINALDO F. PRISCO
0000                  *
0000                  *
0000 11 93 5B                LXI    D,5B93H
0003 21 93 5B                LXI    H,5B93H
0006 1A               NEW    LDAX   D
0007 77                      MOV    M,A
0008 FE 01                   CPI    1
000A CA 04 2A                JZ     2A04H
000D E5                      PUSH   H
000E 47                      MOV    B,A
000F 23               SKIP   INX    H
0010 13                      INX    D
0011 1A                      LDAX   D
0012 77                      MOV    M,A
0013 23                      INX    H
0014 13                      INX    D
0015 1A                      LDAX   D
0016 77                      MOV    M,A
0017 23               HIT    INX    H
0018 13                      INX    D
0019 1A                      LDAX   D
001A 77                      MOV    M,A
001B FE 9A                   CPI    9AH
001D CA 0F 00                JZ     SKIP
0020 FE 0D                   CPI    0DH
0022 CA 4E 00                JZ     UDATE
0025 FE 8F                   CPI    8FH
0027 CA 45 00                JZ     REM
002A FE 22                   CPI    22H
002C CA 39 00                JZ     QUOTE
002F FE 20                   CPI    20H
0031 C2 17 00                JNZ    HIT
0034 2B                      DCX    H
0035 05                      DCR    B
0036 C3 17 00                JMP    HIT
0039 23               QUOTE  INX    H
003A 13                      INX    D
003B 1A                      LDAX   D
003C 77                      MOV    M,A
003D FE 22                   CPI    22H
003F C2 39 00                JNZ    QUOTE
0042 C3 17 00                JMP    HIT
0045 23               REM    INX    H
0046 13                      INX    D
0047 1A                      LDAX   D
0048 77                      MOV    M,A
0049 FE 0D                   CPI    0DH
004B C2 45 00                JNZ    REM
004E 23               UDATE  INX    H
004F 13                      INX    D
0050 E3                      XTHL
0051 78                      MOV    A,B
0052 77                      MOV    M,A
0053 E1                      POP    H
0054 C3 06 00                JMP    NEW

SYMBOL TABLE

HIT    0017    NEW    0006    QUOTE 0039    REM    0045
SKIP   000F    UDATE  004E
```

Listing 1. Machine-language Blank Removal program.

line of BASIC and branches to the NEW (line) routine.

Concluding Remarks

The first time I used the program in Listing 1, I thought something was wrong with it since the BASIC prompt READY appeared almost as soon as it was loaded. But when the BASIC program was listed, all the blanks were gone, just as they were supposed to be. I tested the BASIC program with a RUN and it ran perfectly, as did several others I tested.

It never ceases to amaze me after working in BASIC for a while just how fast a machine-language program seems to run. I hope that you get as much use out of this program as I have. ■

Reader Service—see page 226

Here's an aid for disorganized or beginning computer users who find themselves drowning in paper.

Managing the Small System Environment

By Steven K. Roberts

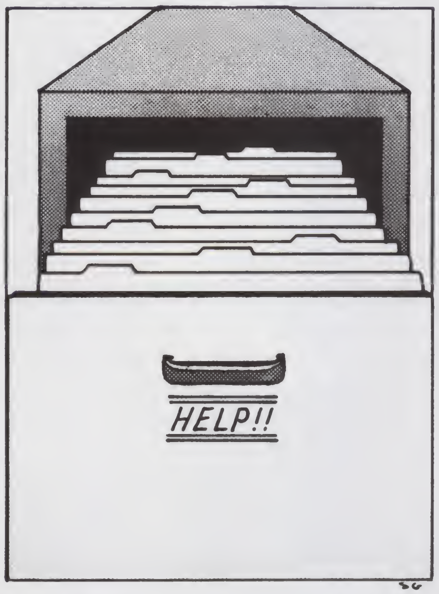
It starts out innocently enough. You clear a space, install a computer and plug it in. You play with it a bit, juggle some I/O drivers, initialize a few disks, bring up an applications package or a language of choice and find yourself with a system.

You are on line.

You possess a powerful tool, a relentlessly educational toy and, if you're not careful, a guaranteed ticket to insanity. Whether your reason for all the effort is business, software development, general utility, music or simply an obsession with computers, you have created a set of environmental requirements that cannot be ignored: from the tiniest microprocessor to the strange world of Amdahl and Cray, computers need organization.

More accurately, people need organization. Two years after implementing a convenient little I/O driver patch, you may change peripherals and need to unpatch it. At 300 revolutions per minute, a minifloppy rotates over 150 million times in a year of continuous operation—it needs replacement and the heads need cleaning. The wiring of a cable, the revision levels of the boards and the hardware patches may all become important when a problem develops.

Files get misplaced. I/O port assignments are forgotten. Data bases have no backups. What happened to the idea of "computer as appliance"?



Unless your system is a sealed black box with a dedicated function, you may be in for some unwelcome surprises—unless, of course, you incorporate into your installation a few techniques of site management. The definition of the term "system" must be expanded to include your environment.

Levels of Organization

Outside the comfort of the card cage and the confines of an operating system, it's a cruel and confusing world. The most elegant and beautifully integrated system architecture can be mired in poor documentation. Too often, the computer manufacturer or software vendor creates problems up front by ignoring this central fact, and a user saddled with such a system is, alas, usually powerless to do anything about it.

But assuming a quality system with reasonable documentation, you are in a position to implement a smooth-running, efficient installation. Attention to six major areas of system organization can make this possible, and make life happier for all concerned.

Hardware documentation—Oddly, this is frequently overlooked despite its almost automatic nature: manufacturers provide manuals, and hardware changes are seldom made without notes. Yet some "minor" details are easy to forget.

System documentation should start with a file cabinet or, at the very least, a dedicated file drawer. The hardware section should begin with records of purchases, warranties

Steve Roberts is a freelance writer and industrial microprocessor systems consultant residing near Columbus, OH. He has been involved with small system design and applications since the early 1970s and is the author of Micromatics *(Scelbi Publications, Elmwood, CT, 1980), as well as about 30 magazine articles. His address is 5885 Dublin Road, Dublin, OH 43017.*

with the registration dates noted and the serial numbers of all equipment. In case of equipment failure or loss, this information can prove suddenly indispensable.

A second hardware file should be for factory support, with the names and phone numbers of knowledgeable people at the various manufacturers represented in the system. This is a good place for merchandise return forms, the names of a few local wizards and names of some other users of the same equipment.

Third, you should have accurate and up-to-date records of all changes and modifications made to the system: cable wiring, modifications to circuit boards and those temporary patches that somehow become permanent. Even such trivial items as a remote reset switch should be documented in the form of a sketch and filed.

A fourth file might be labeled, "Things to Do." There is good reason to avoid frequent power cycles on a computer, and minor changes and additions, such as optional PC board modifications suggested by the manufacturer, can often wait until the next major shutdown. An organized queue of patches can help keep the system configuration up-to-date.

Fifth, you should have a file which documents the current system configuration. What boards are installed? How many spare edge connectors are there? Which memory board corresponds to which address range? What are the I/O port assignments?

This should also include a sketch of the overall system, showing all cables. You'll appreciate a good configuration diagram when the system is moved to a new home.

And then, of course, you might have any number of hardware files detailing the various elements of the system. Kit plans, schematics, theory of operation and any other data relating to specific boards or peripherals should be sorted into convenient subheadings (16KZ memory board, Diablo Printer, Hazeltine 1500, homemade modem, etc.). Again, the easy availability of documentation about the system hardware can save time and frustration when it is time to repair, replace or modify the machine.

Maintenance—Immediately following the hardware section in the file cabinet, there should be a maintenance section. It should contain four files:

Hardware Section	
Purchase Records	Factory Support
Modifications	Things to Do
System Configuration	16KZ Memory Boards
Z-2D Kit Plans	Diablo Printer
Hazeltine 1500	Modem
Analog Hardware	Music Hardware
Home Control System	
Maintenance Section	
Maintenance Log	Questions & Problems
PM Schedule	Media Life Records
Environmental Software Section	
I/O Drivers	Revision Log
Questions & Problems	Memory Map
CDOS	BASIC
ASMB	FORTRAN
DBMS	TRACE
FORMAT	COBOL
LISP	
Applications Software Section	
Music Programs	Corning Job
Graphics	Communications
Games	Home Control System
Payroll	GL
AR	AP
Inventory	Backup Log
Operations Section	
Procedures	Master File Directory
Library Section	
Magazine Indexes	Library Routines
Clippings—Software	Clippings—Hardware

Typical computer room file organization.

First, the system maintenance log. This is important! Every fix, every problem and solution should be noted for future reference. No matter how obvious the solution, it will probably be forgotten 18 months later when the same failure occurs again. A quick scan of the maintenance log might reveal the identical problem and prevent a lot of head-scratching.

Second, you should have a "Question and Problems" list. Periodic conversations with the manufacturer yield more useful information if all of the accumulated questions are in front of you while you talk. It is too easy to forget half the things you wanted to ask.

Third, a preventive maintenance (PM) schedule is a must. Printers need to be oiled, filters and heads need to be cleaned, and analog portions of the system need periodic calibration. These things are normally forgotten in the complacency of smooth system operation, then regretted much later when failure results.

And last, the maintenance section should include the dates of acquisition of the system media—diskettes and cassettes in particular. This alone will not give an index of media use and optimal replacement time, but it will aid in keeping track of its age. Diskettes do fail from extended use—believe me.

Software documentation (environmental)—The environment in which application jobs run and in which development work is accomplished consists of an operating system, a monitor and various resources such as assembler, BASIC, COBOL, data base management system, text editor, text formatter and debugger. Also, you can have miscellaneous custom utilities, I/O drivers different from those provided with the system and macro libraries. All this code can be loosely lumped into the "environmental" category, and most of it is usually documented in the form of manuals. Keep the latest ones around, and put the others in the attic.

Patches to the operating system probably occur most often in the area of communication with I/O devices. A software driver for a daisy wheel printer or a memory-mapped CRT represents a key link between the system and the outside world—the

documentation which defines it should not become buried in the inevitable computer room piles of paper, but should become file number one in the software section.

Like hardware, only changing more rapidly, software is identified by revision numbers. A complete record of these, along with any software registration forms, should be available at a glance. The first question asked by a software vendor when confronted by a request for help is usually, "What revision do you have?"

This file should also note, by diskette number, which revisions of the various pieces of environmental software exist on each piece of media. With 50 diskettes and six different revisions of the operating system, confusion can reign supreme (especially in situations where subsequent revisions are not compatible).

Again, you should have a file of questions and problems for the software supplier. If there is an obscure glitch in BASIC, it should be noted here so that the vendor can be informed and possibly provide a solution.

An important element in your environmental software file is a memory map. This is a drawing of the system's address space showing the regions occupied by the operating system, user RAM space, stack area, ROM locations, the addresses of any memory-mapped I/O devices and so on. When you are adding a patch and looking for a buffer area that will not plague you in the future, this can be a big help.

Software documentation (applications)—The organization of information relating to applications software is very much a function of its volatility, whether you are its user or creator. If you are a user of an applications package, the principles outlined above apply, since it can be considered part of the system environment.

If the system is being used to generate new software, however, we have what could accurately be termed a can of worms. Revision numbers whiz by, sometimes up to the hundreds before the design is frozen and the program is delivered or put to work. This level of activity calls for careful management, both in the system and in the file cabinet.

There should be records, of course, defining the design criteria and the programming philosophies of which the software is comprised, filed by program or, in the case of small jobs (like "Miscellaneous Music Programs"), by category. Sample runs, memory utilization maps, debugging notes and latest listings should be filed together by job.

At the system level, you'll want to reflect the revision number in the filename. Most systems provide the option of simply editing a source file by its existing name, producing a .BAK file as well as the new one, or creating a new name each time and using the old one as the input file. In the case of a complex project, the latter is preferable. In the case of text files and simple programs, the former is adequate. The difference lies in the amount of historical information that may be important in the evolution of the work.

In either case, you should have many backups, scattered across a minimum of three diskettes. The presence of multiple revisions on a single diskette is little comfort when an error message indicates a nonrecoverable media error in the directory—a rare, but crippling, event.

Operations—The area of system operations—the day-to-day running of the computer—is one in which some planning in the early stages of the system's existence can pay off continuously throughout its operating life.

Operator conveniences work wonders. An ASCII code chart on the wall, reference cards near the console and a supplies drawer containing EPROMs, connectors, printer ribbons, typewheels, oil, head cleaning supplies, diskette labels, a forms ruler, razor knife and more felt pens than you'll ever use will make life in the computer room easy. The less you have to run around tracking down trivia, the better.

In the file cabinet, you should retain written procedures for the tasks that have not become second nature; e.g., patching the operating system with new I/O vectors, performing a CONFIG or loading certain debugging utilities. Strange but true, it is possible to forget some of the less frequently used procedures.

If the system has not yet been equipped with a multimegabyte hard disk, the problem of organizing dozens of diskettes must be faced. In the case of minifloppies (5 inch), the organization of an active system can create considerable confusion. This can be minimized by—you guessed it—careful planning.

First, the diskettes should be se-

✓Reader Service—see page 226

quentially numbered (irrespective of their contents) and separated into broad categories. There may be disks for writing and storing BASIC programs, disks for creating text files and disks for data base. The point is to separate them by function; this will eliminate the confusion that can result from having articles, I/O drivers, a COBOL accounting job and miscellaneous BASIC programs on the same diskette.

Second, any critical file should be copied onto one of a number of general backup diskettes, which are stored in a different location from those in daily use. A fire, the catastrophic introduction of a magnet into the room or a careless coffee spill can have dismaying effects if all of the system files are in the same box. If you are doing custom software and possess absolutely irreplaceable data, invest in a safe deposit box. It's cheap insurance.

Third, you can keep track of all those files by periodically preparing a master directory of all of your diskettes and manually updating it as the files evolve. It would be pleasant to dedicate an additional drive to the system, along with appropriate support software, to automatically update a master directory file each time the directory of another disk is changed. This, along with some dedicated data base techniques, would effectively solve the problem of organizing a system environment consisting of large numbers of floppies.

Lacking that convenience, however, generating a complete directory manually at appropriate intervals considerably simplifies the task of finding a long-dormant file.

And fourth, each diskette should carry a brief text file for identification. A convenient way to name this file is IDENT.MYY, where M is a month code (0–9, A and B) and YY is the year of the most recent change in the IDENT file. The file consists of the diskette number, its acquisition date and a brief summary of its use and present function, including any error incidents. This, printed with its directory upon each generation of the master directory file, fills the missing links in the system files' documentation.

Attention to these and similar aspects of system operation can make computer use much more pleasant than it is when confusion reigns. Organizational techniques add little to the operating overhead, but pay off handsomely in convenience.

The computer room library—There are sources of pertinent information that need not be integrated with the active system files, disk and otherwise, that we have discussed so far. These are vendor literature, magazines and books. There are thousands of published programs which together represent a powerful resource to the microprocessor user, along with hardware articles and philosophical comments. How can you possibly use this massive resource effectively?

The fundamental solution is a computer room library, cross-referenced as heavily as your time permits. The problem, of course, is that the sheer volume of information can make this cross-referencing a prohibitive task. Even with the use of a data base management system, time required to update may be too expensive to consider.

An intermediate solution is the creation of another category in the system file cabinet. Use it to save annual indexes from the magazines, notes about articles of particular interest and photocopies or clippings of immediate interest. Published information can be a real time-saver, and it is worth trying to organize it.

Conclusions—If you find yourself plagued by relentless stacks of papers that are too valuable to toss out but too numerous to try to organize, take the time to create a file drawer. Starting with the categories outlined in this article, add files that fit the requirements of your own unique installation and of your work. When you have a definite place to file almost any piece of information, those piles take a lot longer to form.

Until we have self-documenting systems, large cheap disks that back themselves up and hardware and software so perfect that documentation is superfluous beyond the "how to use it" level, we will have problems in organizing the small system environment. These requirements are an established part of the data processing industry, but only with the recent proliferation of microprocessors have they arrived in such force at the doorstep of the small businessman and hobbyist.

But as long as the potential user is equipped with the organizational techniques to deal with this flood of information, the threat posed by computers to his or her sanity can be held at bay.■

Reader Service—see page 226

Real-Time Spectrum Analyzer

By Robert Baker

Eventide Clockworks (265 West 54th Street, New York, NY 10019) offers several accessories for the PET/CBM. Of special interest is their THS224 Real Time 1/3 Octave Audio Analyzer.

Eventide provides all necessary hardware with the board. Power comes from the PET transformer, but not from its regulators. Installation of the analyzer allows rapid and versatile analysis of virtually any audio signal. Additional capabilities include voice recognition, statistical processing and room acoustics. The analyzer does not affect normal operation of the computer.

The analyzer board is mounted inside the PET, with spacers holding the board several inches above the PET main logic board. Two cables then connect the analyzer to the memory expansion connector(s) and the power connector of the PET. Another cable from the analyzer board attaches to a standard stereo jack, with a small metal bracket to mount it on the back of the PET. This connector supplies the audio input signal to the analyzer.

A ROM on the analyzer board provides various routines to control the associated hardware. The ROM occupies locations $B000 to B3FF hex on the PET address space. To save you the bother of memorizing many addresses, the USR function provides the linkage to the PET operating system. By calling various USR functions, the machine-language routines are accessed conveniently by BASIC programs.

On the newer 16K/32K PETs, you must alter a configuration jumper on the PET's main logic board to let the computer access the analyzer. Two jumpers control the selection of the empty ROM sockets during the 9, A and B selection times. By changing these jumpers, the analyzer board is then accessed by these address ranges.

Once these jumpers are changed, however, you cannot use any product that requires a ROM in the normally open ROM sockets (such as Word Pro or the Toolkit). This also prohibits upgrading to BASIC 4.0 ROMs, since there are five ROMs in the new set. Another reason for not being able to upgrade to BASIC 4.0 is that the analyzer uses areas of RAM in the second cassette buffer, which is also used by BASIC 4.0 when processing disk commands.

What Is Real Time?

The Eventide THS224 is classified as a "real-time, constant percentage bandwidth" spectrum analyzer. A real-time analyzer continuously analyzes all components of interest in the input signal and provides information in some usable form. This is in contrast to, for instance, the swept filter type of analyzer, which can analyze only one frequency at any given time.

There are three general methods of performing real-time analysis. The first, as used in this unit, provides a large number of bandpass filters so that all components are analyzed simultaneously. The other methods involve digital capture of signal segments and their processing, either by analog or digital techniques. While the latter two methods are in many cases more powerful, their disadvantages, such as high cost and slow speed, render them unsuitable for applications in which audio spectra must be observed many times per second.

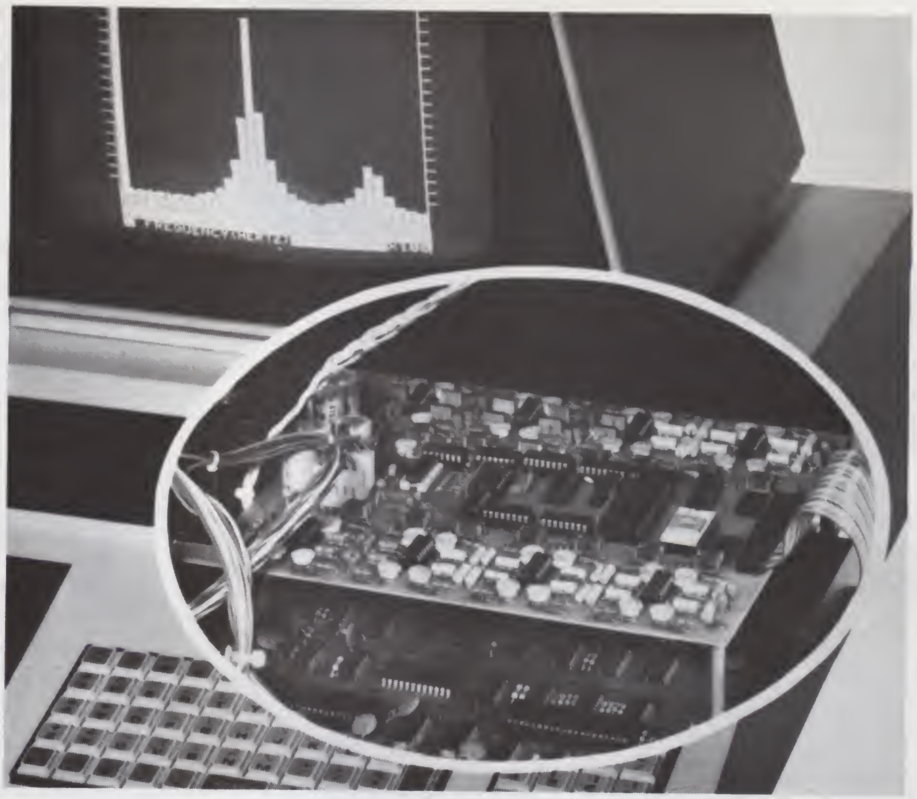
The constant percentage bandwidth consideration also enters the picture. The phrase means that each filter has a bandwidth which, when divided into its center frequency, yields a constant. For instance, if a filter has 3 dB points 2 Hz apart at a 20 Hz center frequency, another filter should have a 3 dB bandwidth of 20 Hz at a 200 Hz center frequency, and so on. This characteristic, fortunately, is almost unavoidably achieved when building filters using identical hardware configurations and different resistive or capacitive values to select the center frequency.

Theoretically, the filter center frequencies are derived by starting with the first frequency desired and multiplying each successive filter center frequency by one-third octave, or the cube root of 2. Since it is desirable to have the filters correspond to decade ratios as well, several standards organizations (ISO/ANSI/IEEE and others) have agreed on a set of center frequencies which nicely rationalize the disparate ratios. The filters used in the Eventide analyzer are two-pole filters with a Q of

Robert Baker, 15 Windsor Drive, Atco, NJ 08004.

10, which determines the bandwidth. They're all identical except for the resistor and capacitor values that determine the center frequencies, and the filter center frequencies conform to the standard values.

Although the input signal could be applied immediately to the filter inputs, it is customary and convenient to provide a variable gain preamplifier. The most important reason for this is to preserve an adequate dynamic range. The filter peak-to-peak output voltage swing is limited by the power supplies to almost 20 volts. The voltage measurement capability of the unit divides the positive half of this swing into 255 even steps. If the input signal were, say, only 2 volts, then only 25 steps could be measured and the accuracy and resolution of the analyzer would be drastically limited.



Eventide's real-time audio spectrum analyzer plugs right into the PET.

Operation

Variable gain is implemented in a somewhat unusual fashion in the Eventide analyzer. A standard operational amplifier in an inverting configuration is used as the "preamp." In the feedback loop of the opamp is a CMOS digital-to-analog converter (DAC) with its input code controlled by the PET. The DAC is of the "four quadrant multiplying" type, meaning that the reference input (connected to the preamp output) is multiplied by the digital word applied to the converter. If the maximum word (255) is applied, the reference is multiplied by one, giving an opamp gain of one; when the input code is 128 the gain is now 2, or 6 dB, and so on.

The gain available from the preamp is sufficient for almost all ordinary signal sources, such as audio consoles, hi-fi "aux" outputs, guitar pickups, etc. It is not sufficient for very low-level signals, such as dynamic microphones, phono cartridges and telephone pickup loops. An external preamp is recommended for this type of application.

Once the input signal is applied to the filters, it is separated into its various frequency components. With the filter outputs being ac signals, they may assume any value between their peak amplitudes. To make a measurement of their absolute values, the "envelopes" amplitude must be measured. This is done with a diode detector much like that used in an AM radio receiver. Whenever the filter output instantaneous amplitude is more positive than the voltage on the capacitor at the output of the filter, the diode conducts and the charge is transferred to the capacitor. The analyzer also has one diode detector that is directly connected to the input signal to measure the level of the composite input signal. In this case, all input signals, regardless of frequency, affect the charge on the capacitor.

To graphically display a spectrum, the various charges on the detector capacitors must be measured and converted to bar heights on the PET display. This is done in three steps:

- Scan the various filter outputs,
- Cause the A/D converter to measure the voltage on the detector capacitor, and
- Store the values in the second cassette buffer areas.

In practice, each of these steps is performed sequentially for each filter before going on to the next. The data collected is then processed by the various ROM routines provided on the analyzer board, however desired by the user software.

As mentioned earlier, the various machine-language routines contained in the analyzer ROM are accessed via USR functions. When first turned on, the PET has no knowledge that the analyzer is there. A specific SYS command must first be issued to initialize the analyzer board and preset the entry address for subsequent USR functions. Then the various USR functions are issued as desired.

USR (0)—DISABLE FUNCTION disables the analyzer and returns the PET to normal operation.

USR (1)—SCALE FUNCTION erases the PET screen and displays the blank graph scales for displaying analyzer data.

USR (2)—BARGRAPH FUNCTION fills in the axes displayed by USR (1). The data from which the graph is derived resides in the second cassette buffer, between memory locations $33A and 35A hex. This function can be used to plot any data contained in these locations, so the routine may be used with other BASIC programs.

USR (3)—SCAN FUNCTION performs the spectrum analysis, reading the amplitude of the input signal as passed by the individual filters. The results of the analysis are placed in the buffer between $33A and 35A as described before. The various modes of operation (logarithmic, linear, single, average, fast and slow) are active during the SCAN operation and directly affect the data placed in the buffer. The effect of each of these modes is described under its particu-

lar USR function later. The return parameter of USR(3) is an integer between 0 and 30, representing the number of the frequency cell with the highest signal level.

USR (4)—SLOW DECAY FUNCTION sets a flag which is accessed during the scan routine. Normally the scan routine takes the absolute amplitude of the data in each frequency cell and transfers it to the buffer location as described earlier. If the decay flag is set, the current amplitude data in any given cell is compared with the previous amplitude of the same cell. If the new data is greater than the old data, the new data replaces the old data. If the old data is greater than the new data, the amplitude is permitted to decrease only by an amount set in location $3A3 hex. The decay rate constant is normally initialized to 6 but may be poked to any desired value.

USR (5)—FAST DECAY reverses the effect of USR(4). The decay flag is reset, and the value of the decay rate constant becomes irrelevant.

USR (6)—AVERAGE FUNCTION places the analyzer in the average mode and zeroes two areas in the second cassette buffer used to store the data developed in the average mode. The first area is the number of averages counter, a double byte counter at locations $3A0–3A1 hex. When the average flag is set, the double byte counter is incremented each time a scan function is activated.

Simultaneously, the data from the various frequency cells is added to corresponding double byte cells in the new buffer area starting at hex location $35D. The data in these locations corresponds to the sum of the amplitudes of the data in the first displayable cell on the screen from the time that USR(6) was originally invoked. The data may be normalized by dividing the current data in each cell by the current number of averages.

USR (7)—DISABLE AVERAGE resets the average flag and prevents additional scans from changing the stored data. The number of averages mentioned above is returned by the function.

USR (8)—LINEAR causes the height of the bargraph created by USR(2) to be precisely equal to the absolute amplitude of the data returned by the scan [USR(3)] process.

USR (9)—LOGARITHMIC SCALE sets the LOG flag which tells the scan function to compute the logarithm of the amplitude in each frequency cell and deposit this value in the buffer. This is done after any averaging is performed and before any variable rate decay function is performed.

If any other USR argument is attempted, an error message will be displayed to indicate an illegal USR call was issued.

Upon initialization, a value of 128 decimal is poked into the gain control port located at 46080 ($B400 hex). This corresponds to a pre-amplifier gain of 6 dB (or a voltage gain of 2). The analyzer gain can be controlled by poking appropriate values into this location. The rule is that the signal gain increases by 6 dB for each division by 2 of the data poked into address 46080. Warning: poking a value of zero will remove all feedback from the operational amplifier. This results in unstable operation that will not damage the analyzer, but the display will be unstable and incorrect.

The analyzer board comes with three programs on cassette, a Mylar overlay for the keyboard and all necessary installation hardware. The programs include diagnostic and calibration routines along with a program called Interactive. This is a general-purpose utility program for ordinary use of the analyzer. It provides control of the various analyzer functions directly from the keyboard. The keyboard overlay identifies specific keys that are used when running the Interactive program. It also provides a handy reference of all USR functions and buffer locations.

Summary

The analyzer is a well-designed accessory for the PET. It comes with a six-month warranty, extensive documentation and several interesting applications notes. The manual even includes a complete schematic, parts list and board layout diagram. It's a valuable and worthwhile accessory that should be of interest to many PET owners. Write to Eventide Clockworks for pricing and further information. Various versions are available to cover all model PETs. Just remember the restrictions on using additional ROMs and the second cassette buffer. ■

✓ Reader Service—see page 226

Computer music synthesis in two parts.

Simulation Of Musical Instruments

By Hal Chamberlin

The use of microcomputers for teaching, composing, transcribing and playing music is rapidly becoming a major application area. New synthesizer boards, music programs, and integrated systems with music capability are among the new products highlighted in the microcomputer field. It is now common for university and even high school music departments to acquire a quantity of microcomputers solely for musical purposes. It is even getting to the point where it is hard to find a microcomputer owner *without* some kind of music program, even if it only plays kazoo-like music through a built-in two-inch speaker.

The Computer Music Field

Since any complete discussion of microcomputer music is impossible within the confines of a magazine format, this article deals with a much narrower subject area. First, however, we need to characterize the field somewhat to see how the topics of musical instrument simulation and software digital synthesis fit in.

Computer music systems cover a broad range of sophistication, application, capability, sound quality and cost. At one extreme, we have the limited-range, tinny, one-voice, "gee whiz" type of system mentioned earlier that can be set up for a dollar's worth of parts and a program whose listing would not even fill a page.

At the other extreme, we have experimental computer music systems in some universities that have a range beyond human perception, quadraphonic sound quality exceeding that of the best recording equipment, virtually unlimited synthesis capabilities and practically infinite voice count at a cost (if measured by industrial standards) in the millions. Using a microcomputer, you can set up a system with reasonably wide range, good stereo sound quality, good synthesis flexibility and 32 voices for a few thousand dollars. The important point is that there is a definite need for systems addressing these extremes and many points in between.

The simple one-voice systems are certainly the most common and are fully adequate for teaching elementary music concepts as well as for impressing friends and neighbors. In fact, they are probably preferred for getting started because their very simplicity makes them easy to learn and use. Since only pitch and timing can be controlled, there are only two variables to worry about. Harmony, timbre, envelope and dynamics are either absent or predetermined.

Note that this type of music system is easily implemented either purely in software using timed loops, which toggle an output port bit, or through a combination of software and hardware where control bytes are sent to a simple divide-by-N counter which may even be part of the I/O interface chip used by the computer. With this level of system, you either quickly outgrow it and move on or are content to file the program alongside the Lunar Lander and Star Trek cassettes.

The next step up is generally either a synthesizer board or an inexpensive eight-bit digital-to-analog converter. The synthesizer board is a set of several oscillators which at a minimum are programmable for pitch and amplitude. (There are a couple of very sophisticated single-voice synthesizer boards, but they are intended to be used in multiples.)

The simplest type of synthesizer board has three square-wave oscillators with pitch and amplitude registers for each and sometimes an overall volume control. Typically, these boards are implemented using programmable timer integrated circuits as the oscillators (normally intended for use in process-control-oriented microcomputers) and discrete circuitry for the volume control function.

Recently, General Instrument introduced a synthesizer chip that has

Hal Chamberlin is vice president of Research and Development for Micro Technology Unlimited, Box 12106, Raleigh, NC 27605. Active in electronic sound synthesis since 1966 and in computer music synthesis since 1970, he has authored numerous magazine articles and has recently published a book entitled Musical Applications of Microprocessors.

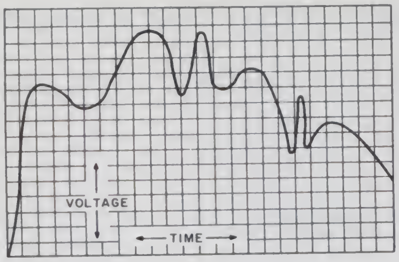


Fig. 1a. Desired audio waveform.

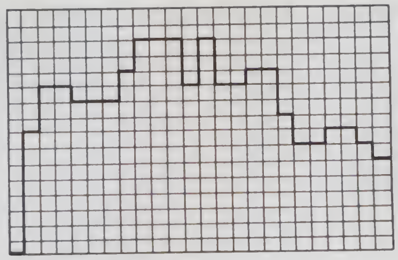


Fig. 1b. Raw DAC output.

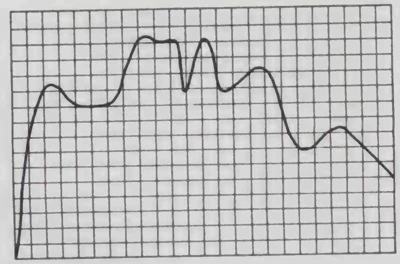


Fig. 1c. DAC output filtered by ideal low-pass filter.

the three oscillators *and* the volume control circuitry integrated on a single chip along with a noise generator useful for limited percussion effects. These chips, usually in trios for a total of nine voices, are appearing in the latest batch of synthesizer boards. Prices range from a little over $100 to nearly $300 with little connection between capability and price.

With these synthesizer boards, the computerist musician gains a great deal of flexibility, since he can play complex chords and control dynamics and tone envelopes. As a result, these boards are a great deal more difficult to master, although you can choose to ignore some of the variables initially.

There is one serious shortcoming, however: all synthesizer boards in this class produce square waves exclusively. (One three-voice board on the market has the capability of combining two of the voices into a single variable width rectangular wave, which increases the tonal variety somewhat.) Square waves have a rather sharp, yet hollow, sound that most closely resembles that of a kazoo. By suitable control of the amplitude envelope, you can produce continuous organ-like tones and percussive plucked-like tones, but the basic character of square waves remains.

With the proliferation of this type of board in recent months (and its constant demonstration at computer shows), the public may very well come to associate square wave sound with computers, just as a piano is associated with its own tone color. This would be unfortunate indeed, since the ultimate value added by computers in music is a wider range of timbres than any other instrument. Nevertheless, there is sufficient expressive power available so that the difference between a piece programmed by a novice and one programmed by an experienced musician is readily apparent.

Music systems based on these synthesizer boards seem to satisfy many users whose goal is to learn music, enjoy transcribing music into the computer and even perform simple composition. They typically will not satisfy a musician attempting to do serious performance work with the computer.

Much more sophisticated synthesizer boards with programmable waveforms are also available in the $500 to $2000 price range. These overcome the lack of tonal variety of the square wave boards by providing programmable waveforms, usually with provisions for a different waveform for each voice.

An important consideration that will be discussed later is whether the board allows dynamically variable waveforms; that is, the ability to smoothly alter the waveform while a note is being played with it. This is a requirement for many effects such as the "wah" of a muted trombone, and, as might be expected, the less expensive units do not provide for it. In either case, the tonal variety is far greater than the square wave units and is sufficient to satisfy many musicians as well as casual users.

On the other side of the hardware/software fence are music systems based on digital-to-analog converters (DAC). As we shall see later, a digital-to-analog converter simply translates numbers into voltages. A very rapid string of numbers produces a rapidly varying voltage; that is, an audio waveform.

In theory, appropriate software can calculate the necessary number sequence to produce literally any sound. The capabilities of a music system based on a digital-to-analog converter are determined solely by the sophistication of the software involved rather than the capabilities designed and frozen into a hardware synthesizer. DAC boards also tend to be less expensive. A good eight-bit DAC board sells for less than $70, while an experimental homebrew unit can be put together for half the price of a movie ticket.

The remainder of Part 1 will describe how sound generation software works in a DAC-based system.

Numbers to Sound

The fundamental principle behind digital sound synthesis is that a string of numbers from a computer program may be converted into a high-fidelity audio signal. As you might expect, the rate at which the numbers are supplied and the precision of the numbers both determine the fidelity of the resulting sound. In synthesis applications, fidelity's usual definition, i.e., faithfulness to the original, does not apply, since there is no original. Instead, fidelity is used to refer to the frequency range that can be produced and the relative freedom from undesired noise and distortion.

Fig. 1 shows how a DAC can produce a smooth audio waveform from a string of numbers and the errors involved. The grid in the figures represents time in the horizontal direction and voltage in the vertical direction. Fig. 1a shows a greatly magnified drawing of a small portion of a typical audio waveform. Notice that it wiggles and curves through the figure without regard for the grid. In Fig. 1b we have the raw output of a DAC being fed the string of numbers representing the waveform in Fig. 1a.

Each vertical grid line represents the point in time that the DAC receives a new number; thus, it stands to reason that the DAC output can only change up or down at vertical grid lines. Each horizontal grid line represents a possible numerical value that the DAC can receive. For example, an eight-bit DAC can only accept 256 (2^8) different numbers, so the complete grid for such a DAC would have 256 horizontal grid lines. As a result, the DAC output can only dwell at a horizontal grid line. Needless to say, the smoothly curved waveform of

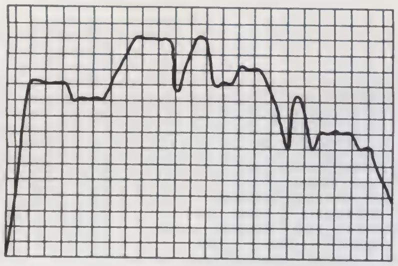


Fig. 1d. Filtered DAC output at twice the sample rate.

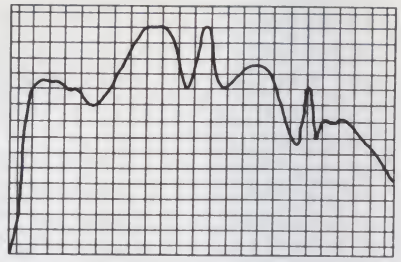
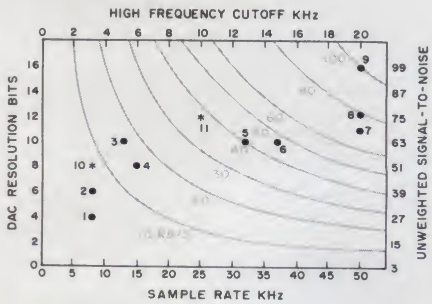


Fig. 1e. Filtered DAC output at twice the resolution and sample rate.

Fig. 1a is severely distorted in shape by this two-dimension quantization imposed by the DAC.

In Fig. 1c we have passed the raw DAC output through an ideal low-pass filter. Usually such a filter is described as allowing all frequencies below a certain cutoff frequency to pass through while blocking those above it. Here, however, we are using it to smooth the DAC output into something that more closely resembles Fig. 1a.

As you can see, the filter does a nice job of smoothing the curve between the vertical grid lines (that is, between new numbers from the computer). Closer examination, however, reveals that when the curve crosses a vertical grid line, it is also precisely on a horizontal grid line as well. Thus, since it is still constrained by the horizontal grid, there is still some error remaining. The only way to reduce this error is to make the horizontal grid lines denser by adding significant bits to the numbers and the DAC.



1. POOR LONG DISTANCE TELEPHONE
2. GOOD LONG DISTANCE TELEPHONE
3. GOOD A-M RADIO
4. $ 50 PORTABLE CASSETTE
5. $ 200 HI-FI CASSETTE
6. GOOD F-M RADIO
7. CONSUMER REEL-TO-REEL TAPE
8. PROFESSIONAL REEL-TO-REEL TAPE
9. PROFESSIONAL DIGITAL AUDIO
10. REAL-TIME MICROCOMPUTER DIGITAL SYNTHESIS
11. DELAYED PLAYBACK MICROCOMPUTER DIGITAL SYNTHESIS

Fig. 2. Required DAC resolution, sample rate and data required for varying degrees of fidelity.

Comparing Fig. 1c with Fig. 1d, it should be apparent that the horizontal spacing of vertical grid lines determines the shortest waveform wiggle that can be reproduced. Since the horizontal direction is time, this is the same as saying that the horizontal spacing limits the highest audio frequency that can be reproduced. In particular, it takes at least two grid lines to reproduce one cycle of a sine waveform, so it follows that the highest audio frequency must be equal to or less than one-half of the rate at which numbers are fed to the DAC. Actually, you can go this high only when an ideal low-pass filter is used to do the smoothing.

With a practical low-cost filter, audio frequencies should be kept to 40 percent or less of the DAC update, or sample rate. If higher frequencies are attempted, the filter cannot properly smooth the waveform and an unpleasant distortion called alias distortion occurs. Note that there is no low-frequency limit when producing sound with a DAC; you can go clear down to dc if desired.

No matter how dense the vertical grid lines become, the waveform accuracy is always constrained by the horizontal grid line spacing. This constraint leads to background noise and distortion that cannot be filtered out.

Fig. 1e illustrates the effect of adding another bit of precision to the DAC and the numbers it receives. In terms of audio noise level, adding the bit reduces noise by six decibels, a significant but not dramatic amount. In most cases the noise itself is basically white, resembling somewhat the sound of ocean surf.

Fig. 2 shows how the fidelity of various combinations of sample rate (horizontal grid spacing) and DAC resolution (vertical grid spacing) compares with familiar audio devices. Also shown are contours of constant data rate (total kilobytes per second) to give an idea of required system speed.

The important points are that frequency range and background noise level are independently adjustable system parameters and that greater fidelity is accompanied by a higher data rate. Note that it is considerably less expensive in terms of data rate to reduce the noise level than it is to increase the high-frequency limit. The two stars in Fig. 2 represent the two software digital music synthesis systems that will be discussed in this article.

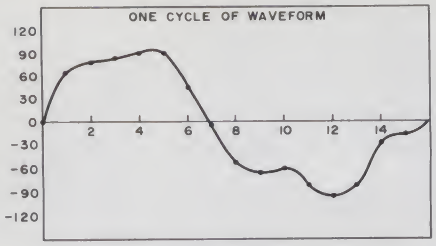
Where the Numbers Come From

The real trick in a DAC-based music system, then, is to compute the string of numbers, or samples, representing the desired sound and then send it to the DAC at the required rate. In all of the cases that will be considered here, the sample rate will be constant because that assumption greatly simplifies the computations. Conversely, when the rate is assumed to be constant, it must be to rather close tolerances to avoid excessive jitter noise.

At this point you can choose to go in either of two directions. In real-time digital synthesis, the samples are computed at the rate required by the DAC and sent to it immediately. The advantage, of course, is that the sound is heard in its final form as the program is running. The disadvantage is that practical sample rates are relatively high, which means that a very efficient program using an uncomplicated synthesis technique running on a fast microcomputer is required.

The other choice is delayed playback digital synthesis, where the computed samples are first written onto a mass storage device at relatively low speed and then later reread and played through the DAC at the necessary high speed. The advantages here are that the synthesis program can be more accurate (and thus slower), any synthesis technique of any complexity can be utilized, and the higher sample rates and DAC resolutions necessary for high fidelity can be utilized. The main disadvantages are a rather long delay between program execution and audible results and the need for a large capacity, high-speed mass storage system.

It is also possible to combine the two philosophies—real time for composition and experimentation with the orchestration and delayed playback for a high-fidelity final result.



WAVEFORM TABLE

ADDRESS	CONTENTS
0	0
1	67
2	79
3	81
4	92
5	93
6	46
7	-7
8	-55
9	-64
10	-60
11	-80
12	-93
13	-86
14	-27
15	-14

Fig. 3. Example waveform table.

The emphasis in this article will be on the real-time type of system, since that is of interest to most people at this time.

It should be noted that the software techniques that will be described actually perform the same functions as the hardware programmable waveform synthesizer boards mentioned earlier. In fact, a study of the history of computer music reveals that the techniques were developed first in software and then later implemented in hardware when it became practical to do so.

Let's begin by choosing a microprocessor, a sample rate and a DAC word size and then determine what can be done within those constraints.

The microprocessor will be a 6502 because of its usage in a large number of computers (PET, Apple II, Atari, KIM-1, SYM-1, AIM-65, OSI and Mattel, to name a few) and its effective high speed in this application (at a standard 1 MHz, approximately 60 percent faster than a 2 MHz 8080 or Z-80). About the lowest sample rate of interest is 8 kHz. This would allow audio frequencies up to a little beyond 3 kHz, which is just below the highest note on the piano keyboard. More important, it might limit the high harmonics of many instrumental sounds and thus give them a somewhat muffled character. Nevertheless, it is high enough to be useful.

Needless to say, the DAC word size will be eight bits because of the microprocessor chosen. This would give an audible but acceptably low background noise level with a DAC having true eight-bit linearity and a good low-pass filter.

Using the 8 kHz sample rate means that the time between samples is a mere 125 us, or around 40 machine-language instructions per sample. Clearly you cannot write the program in BASIC, use the SIN function to compute samples, use floating-point arithmetic or even perform a simple multiply operation and get a sound sample computed in 125 us, much less several for multi-part harmony. Although there are simple (and thus fast) methods of computing sawtooth, triangle and square wave samples directly, only three different waveforms would not be very much variety.

The answer is to compute the waveform ahead of time and put the results in a waveform table. The music playing program then merely looks up the waveform samples in the table—an operation that takes almost no time on a 6502—and sends them to the DAC. Since the waveform is precomputed, even a very slow BASIC program is acceptable for computing it.

A waveform table is nothing more than a string of bytes in memory where each byte is a sample along the stored waveform. For simplicity, which means speed, waveform tables will be made 256 bytes long, or one memory page. The 256-byte content of the waveform table represents exactly one cycle of the stored waveform. Fig. 3 aids in understanding the relation between a waveform and its image in the table. For illustration purposes the table length is assumed to be 16 bytes, but the same principles hold for the 256-byte length.

Now let's see how a music synthesis program might look up, or scan, the waveform table to produce sound. In its simplest form, scanning consists of reading the first entry, sending it to the DAC, reading the second entry, sending it to the DAC, etc., in a loop. When the end of the table is reached, it is necessary to wrap around to the beginning for the next cycle of the waveform.

If you did this at an 8 kHz rate with a 256 entry table, approximately 31 scans would be performed each second, which corresponds to 31 waveform cycles per second, a very low frequency note indeed. For higher frequency notes the scanning could be altered so that every other entry was read, which would give 62 Hz, every third for 93 Hz, etc. Although not intuitively obvious, skipping

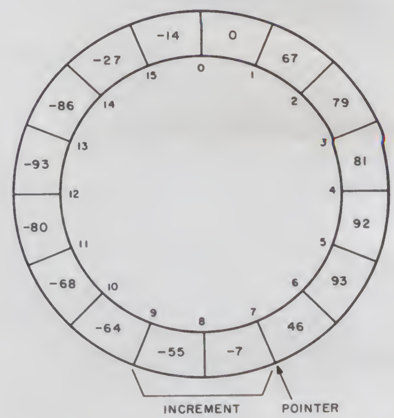
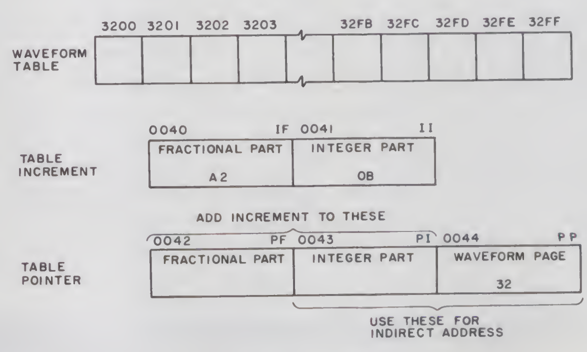


Fig. 4. Waveform table scanning.



ADDRESS	CODE	ASSEMBLY STATEMENT			EXECUTION TIME
0200	A5 42	LDA	PF	ADD FRACTIONAL PART OF INCREMENT TO FRACTIONAL PART OF POINTER	3
0202	65 40	ADC	IF		3
0204	85 42	STA	PF		3
0206	A5 43	LDA	PI	ADD INTEGER PART OF INCREMENT TO INTEGER PART OF POINTER WITH CARRY	3
0208	65 41	ADC	II		3
020A	85 42	STA	PI		3
020C	B1 43	LDA	(PI),Y	DO THE TABLE LOOKUP REGISTER Y=0, A HAS TABLE ENTRY WHEN DONE	5
					23

Fig. 5. Table scanning on the 6502.

waveform table entries does not contribute to distortion if the tabulated waveform conforms to certain rules that will be discussed later.

Fig. 4 aids in understanding the scanning process. Here the example 16-point waveform table has been bent into a circle, which is one way to view the wrap-around process mentioned earlier. The arrow represents the waveform table pointer, which contains the contents of a machine register or memory location. The bracket represents the value of the waveform table increment, which indicates how far the table pointer is advanced every 125 us sample period.

Thus, if the increment is one, the pointer will take on values of 0, 1, 2, . . . , 14, 15, 0, . . . (0–255 in real life) and give us a low note. If the increment is 3, the pointer will go through the sequence 0, 3, 6, 9, 12, 15, 2, . . . and give us a three-times-higher note. Thus, the increment is proportional to the pitch of the synthesized tone. Note that in this case successive trips around the table are not exactly the same. Again, this does not lead to distortion if the waveform meets certain requirements.

Returning to the real case of a 256 point table, it is apparent that the frequency resolution of 31 Hz when using integral waveform table increments is not sufficient for most musical applications. What is needed is the ability to specify an increment with a fractional part such as 7.04 to produce a precise A below middle C. This is quite possible but requires that the waveform pointer also take on a fractional part, which leads to a problem. How should the table be read when the pointer says "read the 78.645th entry"?

A sensible answer would be to look at both the 78th and 79th entries and then interpolate between them. Unfortunately, even simple linear interpolation is fairly complex (requires a multiply), which means it is slow. For real-time digital synthesis on a microcomputer, we will be forced to ignore the fractional part of the pointer when reading the table but include it when adding the increment to compute the next value of the pointer. Taking this shortcut leads to a distortion called interpolation noise, which is significant but generally tolerable.

Now how might a program segment be set up to manipulate the pointer, increment and table to generate sample values for the DAC? Fig. 5 shows the arrangement of a waveform table, its pointer and its increment in memory. For illustration purposes, the waveform table is assumed to be in memory from 3200–32FF, which is page 32, while the pointer and increment are kept in memory page zero for fast access. The increment is a two-byte value with an integer byte and fraction byte as mentioned above. The decimal equivalent of the increment value shown is 11.633. The pointer is actually a three-byte value.

The most significant byte is the page number (32) of the waveform table and normally remains constant but can be changed to select a different waveform. The middle byte is the integer byte of the pointer into that table, while the least significant byte is the fractional part of the pointer.

Every sample period (125 us) the increment is double-precision added to the integer and fractional parts of the pointer, and the pointer is replaced with the result. Any overflow is simply ignored, since it is merely an indication of wrap-around from the end to the beginning of the waveform table. Actual table lookup is extremely simple in the 6502; you simply use the rightmost two bytes of the pointer (the waveform table page address and the integer part of the pointer) as the indirect address of an indirect load instruction. Thus, only one instruction is needed to look up in the waveform table. The 6502 machine-language code shown requires only 23 us to do all of this.

Since the 23 us figure is considerably less than the 125 us allowable, you can have several waveforms, pointers and increments for several simultaneous tones. There is enough time to handle four tones with some left over for housekeeping. You could also have fewer voices and a higher sample rate, or more and a lower rate.

There are two ways to combine the

```
A00F         DAC      =      X'A00F         ; OUTPUT PORT ADDRESS WITH DAC
3000         WAV1TB   =      X'3000         ; WAVEFORM TABLE FOR VOICE 1
3100         WAV2TB   =      X'3100         ; WAVEFORM TABLE FOR VOICE 1
3200         WAV3TB   =      X'3200         ; WAVEFORM TABLE FOR VOICE 1
3300         WAV4TB   =      X'3300         ; WAVEFORM TABLE FOR VOICE 1

0000                  .=     0              ; STORAGE STARTS AT PAGE 0 LOCATION 0

0000 00      V1PT:    .BYTE  0              ; VOICE 1 WAVE POINTER, FRACTIONAL PART
0001 00               .BYTE  0              ; INTEGER PART
0002 30               .BYTE  WAV1TB/256     ; WAVEFORM TABLE PAGE ADDRESS FOR VOICE 1
0003 00      V2PT:    .BYTE  0              ; SAME AS ABOVE FOR VOICE 2
0004 00               .BYTE  0
0005 31               .BYTE  WAV2TB/256
0006 00      V3PT:    .BYTE  0              ; SAME AS ABOVE FOR VOICE 3
0007 00               .BYTE  0              ;
0008 32               .BYTE  WAV3TB/256
0009 00      V4PT:    .BYTE  0              ; SAME AS ABOVE FOR VOICE 4
000A 00               .BYTE  0
000B 33               .BYTE  WAV4TB/256

000C 0000    V1IN:    .WORD  0              ; VOICE 1 INCREMENT (FREQUENCY PARAMETER)
000E 0000    V2IN:    .WORD  0              ; VOICE 2
0010 0000    V3IN:    .WORD  0              ; VOICE 3
0012 0000    V4IN:    .WORD  0              ; VOICE 4

0014 00      DUR:     .BYTE  0              ; DURATION COUNTER
0015 B6      TEMPO:   .BYTE  182            ; TEMPO CONTROL VALUE, TYPICAL VALUE FOR
                                            ; 4:4 TIME, 60 BEATS PER MINUTE, DURATION
                                            ; BYTE = 48 (10) DESIGNATES A QUARTER NOTE

             ;        4 VOICE PLAY SUBROUTINE
             ;        ENTER WITH VARIOUS TABLE POINTERS ALREADY SET UP
             ;        LOOPS TEMPO*DUR TIMES
0300                  .=     X'0300
0300 A000    PLAY:    LDY    #0             ; SET Y TO ZERO FOR STRAIGHT INDIRECT
0302 A615             LDX    TEMPO          ; SET X TO TEMPO COUNT
                                            ; COMPUTE AND OUTPUT A COMPOSITE SAMPLE
0304 18      PLAY1:   CLC                   ; CLEAR CARRY
0305 B101             LDA    (V1PT+1),Y     ; ADD UP 4 VOICE SAMPLES
0307 7104             ADC    (V2PT+1),Y     ; USING INDIRECT ADDRESSING THROUGH VOICE
0309 7107             ADC    (V3PT+1),Y     ; POINTERS INTO WAVEFORM TABLES
030B 710A             ADC    (V4PT+1),Y     ; STRAIGHT INDIRECT WHEN Y INDEX = 0
030D 8D0FA0           STA    DAC            ; SEND SUM TO DIGITAL-TO-ANALOG CONVERTER
0310 A500             LDA    V1PT           ; ADD INCREMENTS TO POINTERS FOR
0312 650C             ADC    V1IN           ; THE 4 VOICES
0314 8500             STA    V1PT           ; FIRST FRACTIONAL PART
0316 A501             LDA    V1PT+1
0318 650D             ADC    V1IN+1
031A 8501             STA    V1PT+1         ; THEN INTEGER PART
031C A503             LDA    V2PT           ; VOICE 2
```

Listing 1. Core sound generation routine for organ-like music.

four table-lookup values into a single eight-bit value for the DAC. One is to simply add them up and send the sum to the DAC, which is the equivalent of audio mixing. When this is done the waveform table values must have been adjusted when the table was computed to avoid overflow (which can lead to horrendous distortion) when the four voices are added up.

The other method is to immediately send each value to the DAC when it is found and let the low-pass filter smear them together, thus effecting mixing. One disadvantage of this approach is that the dwell time of each voice in the DAC must be the same or there will be differences in loudness among the voices. Another disadvantage is that certain DAC distortions are accentuated, although they are usually not significant at the eight-bit level. It is also a simple matter to have two DACs and direct two voices to each for an approximation to stereo.

Listing 1 shows the core sound generation routine used in a digital synthesis program first published in 1977. It is capable of generating four tones simultaneously, where each tone can use a different waveform table. It uses the "add-em-up" technique of mixing the four voices into a single sample value for the DAC. A separate routine is expected to store the appropriate values in each of the four increments for the desired pitches and also set TEMPO and DUR for the desired duration of the chord.

Each time through the main loop takes 115 us and represents one sample period, thus the sample rate is 8.7 kHz. Also, each time through the loop decrements a copy of TEMPO, which is held in the X register. When X decrements to zero, it is restored from TEMPO, and DUR is decremented directly in memory. If DUR also decrements to zero, the chord is complete and a return to the setup routine is taken. Thus, the total chord duration is proportional to the product of TEMPO and DUR. This property makes it possible to change the speed of the music without recoding it.

Note the presence of time-equalizing instructions at TIMWAS so that the loop time is the same whether or not register X decrements to zero. This is necessary to eliminate jitter distortion mentioned earlier.

The setup routine would look at coded music in memory to determine what successive values of the four increments, DUR and possibly TEMPO should be to produce the desired music. Typically, music data would be set up in memory as a set of five bytes for each musical "event" (note or chord) in the piece. The first byte would be the duration, while the other four would represent the desired pitch of each of the four voices.

A note frequency table would be used to determine the proper two-byte value of the increment from the one-byte pitch code. This routine must also be as fast as possible because sound generation is stopped when it is in control. If the flow of samples is stopped for too long, an objectionable click between notes is introduced. See references for a further explanation of the setup routine.

Next month we will continue our discussion of synthesizing multiple tones using waveform table data, explore the capabilities of existing DAC software and examine some of the prospects for the future.■

Last month in Part 1 we examined the hardware requirements of the SN76489A sound chip. Let's now look at the software, with listings for the 8080/Z-80.

Computer Music The Easy Way

By Steve Marum

Last month I discussed the SN76489A computer-controlled sound chip from TI and how to attach it to an S-100-based machine. In this article, I'll cover the software necessary to produce good-sounding music.

Improving the Sound

The problem with the last two programs in part 1 was that BASIC was too slow to produce natural-sounding music. It takes two lines of program to change a frequency, which produces delays of ten milliseconds or more between notes. Although this doesn't sound that bad, it is noticeable when listening to the results. While you need to have a slight pause between consecutive notes of the same frequency so you can tell them apart, notes of differing frequencies sound better if they're run together.

A machine-language program would have no such problem. Data bytes can be sent to the 76489A chip

Steve Marum, 520 Talley, Sherman, TX 75090.

```
0000                   ******
0000                   *  THIS SUBROUTINE TAKES DATA FROM RAM AND SENDS
0000                   *  IT TO THE 76489 CHIP ADRESSED AT 0E800H.
0000                   *  CALL IT AT 0A000H (40960 DECIMAL).
0000                   *
0000                   *  DATA STARTING AT 'DATATABLE' IS SENT TO THE 76489
0000                   *  UNTIL A BYTE CONTAINING 01XXXXXX IS ENCOUNTERED
0000                   *  IF XXXXXX IS ZERO, THE PROGRAM 'RETURNS'.
0000                   *  IF XXXXXX IS NON-ZERO, THE PROGRAM PAUSES FOR
0000                   *  XXXXXX UNIT DELAYS, THEN CONTINUES SENDING DATA.
0000                   *  THE LENGTH OF THE UNIT DELAY IN MILLISEC. SHOULD
0000                   *  BE PLACED IN DE BEFORE CALLING, AND MUST BE <256
0000                   ******
0000                          ORG 0A000H
A000                   DELAY1MS EQU 166
A000                   DELAY30US EQU 20
A000                   PVE EQU 0E800H
A000 21 3B A0          PLAYSONG LD HL,DATATABLE
A003 23                GETNEXT INC HL
A004 7E                        LD A,(HL)  GET NEXT BYTE
A005 47                        LD B,A
A006 E6 C0                     AND 0C0H
A008 FE 40                     CP 040H
A00A CA 1A A0                  JP Z,WAIT  STOP READING IF 01XXXXXX
A00D 78                        LD A,B
A00E 32 00 E8                  LD (PVE),A  SEND DATA TO PVE
A011 06 14                     LD B,DELAY30US
A013 05                LOOP1 DEC B
A014 C2 13 A0                JP NZ,LOOP1
A017 C3 03 A0                JP GETNEXT
A01A 78                WAIT LD A,B
A01B FE 40                  CP 040H  RETURN IF 40H
A01D C8                     RET Z
A01E E6 3F                  AND 03FH
A020 32 3B A0          LOOP2 LD (COUNTER),A NO. OF MIN. NOTES TO DELAY
A023 53                      LD D,E  DELAY OF MIN. NOTE, IN MS.
A024 01 A6 00          LOOP3 LD BC,DELAY1MS  DELAY OF 1 MS.
A027 0B                LOOP4 DEC BC
A028 78                      LD A,B
A029 B1                      OR C
A02A C2 27 A0                JP NZ,LOOP4  GO BACK, MILLISECOND NOT OVER
A02D 15                      DEC D
A02E C2 24 A0                JP NZ,LOOP3  GO BACK, MIN. NOTE NOT OVER
A031 3A 3B A0                LD A,(COUNTER)
A034 3D                      DEC A
A035 C2 20 A0                JP NZ,LOOP2  GO DELAY MORE MIN. NOTES
A038 C3 03 A0                JP GETNEXT
A03B 00                COUNTER DEFB $-$
A03C                   DATATABLE EQU COUNTER  LOCATION IMM. AHEAD OF DATA
PROGRAM IS 60 BYTES LONG (1) WITH 0 ERRORS DETECTED.
> SYMBOL LISTING (Y=YES,N=NO)?
COUNTER=A03B DATATABLE=A03B DELAY1MS=00A6 DELAY30US=0014 GETNEXT=A003
LOOP1=A013 LOOP2=A020 LOOP3=A024 LOOP4=A027 PLAYSONG=A000 PVE=E800
WAIT=A01A
```

Listing 1. Machine-language program used with the EDIT2 BASIC program.

faster than it is capable of accepting them. This means all the registers on the 76489A can be changed in less than 200 microseconds.

One approach to this program—and the best from the user's point of view —is to write the complete software system in machine language. In addition to the play music routine, you must have an editor for entering the notes, a translator to convert the notes to codes suitable for the 76489A to digest and a few bookkeeping routines to load and save songs after they are keyed in. The net result is a quick-running program at the expense of much assembly-language coding.

A Compromise Approach

Not finding the performance of the BASIC program satisfactory and not being ready to tackle a complicated assembler program, I opted for a compromise between the two approaches. The entire program doesn't need to run fast to produce proper-sounding music. Only the routine that actually plays the music needs to run at full microprocessor speeds.

The translation of the music from music notation to 76489A-style codes can be done at a more leisurely pace if the results are stored for later playback. Thus, I have a BASIC program that contains the editor, translator and bookkeeping, plus a machine-language program that plays the translated music.

These two programs are shown in Listings 1 and 2. Listing 1 is the assembler program, using Zilog mnemonics on the XL-Z80 Software Development System, Version 2.1. Listing 2 is the BASIC program written in North Star BASIC Version 6, Revision 5, Release 4.0. Although Z-80 mnemonics are shown in Listing 1, I've used only the 8080 subset of op codes to facilitate transferring this routine to other computers.

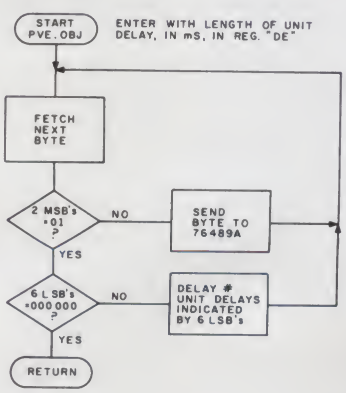


Fig. 1. Machine-language subroutine flowchart to drive SN76489A. To play music on the sound generator board, a simple machine-language program is used to rapidly update the registers on the sound IC between notes. This routine uses a table of data previously stored by a BASIC program, which did all of the complicated processing. All that's left to do is send the bytes to the music board and pause between notes to allow them to be heard.

The Assembly-Language Program

Fig. 1 shows a flowchart of the assembly-language program. As you can see, it's a simple program. It transfers bytes from memory to the 76489A circuit and pauses occasionally. The data to be sent to the 76489A must be loaded into the memory starting with the location immediately following COUNTER. From looking at the table of data formats used by the 76489A, you can see that in bytes beginning with 0 the second bit is a "don't care." I've left this bit a 0 in regular data bytes and set it to 1 when I want to flag this particular byte as a time delay value. When the machine-language program comes to a byte beginning with 01, the remaining bits are used to indicate the amount of time to waste.

You may have to change several bytes to use this program on other systems. The value of DELAY30US must be large enough so that the time it takes to get from GETNEXT back around the loop to GETNEXT again is greater than the pulse width in one-shot Z59. The value of DELAY1MS is chosen so LOOP4 takes one millisecond to count it down. PVE must point to the address of the 76489A. If you relocate the routine, all the absolute addresses must be adjusted accordingly.

The BASIC Program

Listing 2 is the BASIC program that edits and translates the music into byte-sized pieces for the 76489A. It is written in a somewhat structured for-

Listing 2. EDIT2, the North Star BASIC program that translates from music notation to codes suitable for the music generating board.

```
10 REM  3-VOICE MUSIC EDITOR/TRANSLATOR  7-26-79
15 REM  THIS PROGRAM PREPARES CODE FOR AN SN76489 HAVING A 2MHz CLOCK
20 DIM N$(6060),I$(6),T(14),F$(10),C1$(28),C2$(28),B$(60),W(6),L(3)
21 DIM N1$(3),T1(3),V(3),O(3)
200 REM  *** INITIALIZE ***
205 REM  SET LOOKUP TABLES
210 FOR J=1 TO 14\READ T(J)\NEXT J
215 DATA 1012,956,902,851,804,758,716,676,638,602,568,536,506,478
225 C1$="CbC C#D D#E F F#G G#A A#B B#"
230 C2$="CbC DbD EbE F GbG AbA BbB B#"
235 REM  CLEAR NOTE ARRAY
240 B$=""\FOR J=1 TO 60\B$=B$+CHR$(0)\NEXT J
245 FOR J=60 TO 6060 STEP 60\N$(J-59,J)=B$\NEXT J
250 REM  SET DEFAULT VALUES
255 P=1\S=50\N=59392\F1=0\F2=1\F3=0\P2=40960
260 FILL N,159\FILL N,191\FILL N,223\FILL N,255\REM TURN OFF PVE

1000 REM  *** INPUT COMMAND FROM TERMINAL ***
1010 INPUT "Command:  ",I$
1020 IF I$="PLAY" THEN 2000
1030 IF I$="REPLAY" THEN 3000
1040 IF I$="TEMPO" THEN 4000
1050 IF I$="GOTO" THEN 5000
1060 IF I$="INPUT" THEN 6000
1070 IF I$="INSERT" THEN 7000
1080 IF I$="DELETE" THEN 8000
1090 IF I$="CLEAR" THEN 9000
1100 IF I$="SAVE" THEN 10000
1110 IF I$="LOAD" THEN 11000
1120 IF I$="LIST" THEN 12000
1130 IF I$="FSAVE" THEN 13000
1140 IF I$="FLOAD" THEN 14000
1150 PRINT "Unrecognizable command. Try another ",\GOTO 1010

2000 REM  *** TRANSLATE SONG, THEN PLAY IT ***
2015 V(1)=15\V(2)=15\V(3)=15\REM ALL TG'S OFF
2018 O(1)=0\O(2)=0\O(3)=0\REM ALL TG'S OFF
2020 L(1)=-99\L(2)=-99\L(3)=-99\REM ALL TG'S UNUSED
2025 P1=41020\REM 1ST BYTE OF DATATABLE
2030 FOR J=6 TO 6000 STEP 6
2035 FOR K=1 TO 6\W(K)=ASC(N$(J-6+K,J-6+K))\NEXT K
2040 IF W(1)<>255 THEN 2050
2045 S=W(2)\GOTO 2280\REM SET TEMPO
2050 FOR K=1 TO 3
2055 IF W(2*K-1)=0 AND L(K)<>0 THEN 2155
2056 IF W(2*K-1)>0 THEN 2060
2057 L(K)=-99\FILL P1,112+32*K+15\P1=P1+1\GOTO 2155
2060 IF W(2*K-1)<>128 THEN 2085
2065 V(K)=W(2*K)
2070 IF L(K)<0 THEN 2080
2075 FILL P1,112+32*K+V(K)\P1=P1+1
2080 GOTO 2155
2085 IF W(2*K-1)<>48 THEN 2089
2086 FILL P1,112+32*K+15\P1=P1+1\O(K)=0\GOTO 2145
2089 IF L(K)>=0 AND O(K)=1 THEN 2095
2090 FILL P1,112+32*K+V(K)\P1=P1+1\O(K)=1
2095 K1=INT(W(2*K-1)/16)\K2=W(2*K-1)-16*K1
```

More

mat using the top-down approach. The "mainline" is just a short section that asks which function is to be performed. The actual processing of that function is done in a separate routine selected by the mainline. The print statements are written to use an SWTP CT-82 terminal. You may have to eliminate some of the control codes (such as form feed, 0C hex) if your terminal won't handle them.

Lines 10–260 initialize the arrays and lookup tables. After that, all four outputs of the 76489A are turned off. The 76489A can make some pretty raucous sounds when powering up; you may want to move this line closer to the beginning of the program. (Better yet, write an initialization routine that will turn off the 76489A every time your computer is turned on.)

Lines 1000–1150 look for a valid command to be typed in, then branch out to the proper section of code. Any new functions you may add can be handled by inserting more tests between lines 1010 and 1150. Writing the program in this manner greatly simplifies debugging.

Once you get to line 1150, you have a complete program that will run. You'll get an invalid line number error if you try to run one of the functions. However, after typing in two or three routines, you can execute the program and get them debugged before other, perhaps more complex, sections are written.

I started out with the GOTO, input, insert and delete routines and got them going before I went on. By the time I got to save and load, I already knew I could get valid data into the memory from sheet music. Then, assured that all my test data was safely on disk in case the program crashed (which it did—several times), I tackled the play routine. This is definitely the way to write a complex program.

Before going further, I should make a few comments on the general flow of the program. As music is typed in from the keyboard, it is partially translated and saved in a large string array a byte at a time. After the octave and note are entered, they are crunched into one byte. The duration, which is entered next, is also converted to a single byte. Thus, all the information necessary to specify three notes is contained in six bytes, and a string 6000 bytes long will hold 1000 notes. This is more than enough for most songs; the most I have used to date is about 650 notes.

It is important in multiple-voice

Listing 2 continued.

```
2100 K3=T(K2)
2105 IF K1=2 THEN K3=K3/2
2110 IF K1=3 THEN K3=K3/4
2115 IF K1=4 THEN K3=K3/8
2120 IF K1=5 THEN K3=K3/16
2125 K3=INT(K3+.5)
2130 K4=INT(K3/16)\K5=K3-16*K4\REM K4=6 MSB S, K5=4 LSB S
2135 FILL P1,96+32*K+K5\P1=P1+1
2140 FILL P1,K4\P1=P1+1
2141 N1$(K,K)=CHR$(W(2*K-1))
2145 L(K)=W(2*K)
2146 IF L(K)>127 THEN T1(K)=1 ELSE T1(K)=0
2150 IF L(K)>127 THEN L(K)=L(K)-128
2155 NEXT K
2160 K1=99\FOR K=1 TO 3\REM START OF `COUNTDOWN´
2165 IF L(K)>0 AND L(K)<K1 THEN K1=L(K)\NEXT K
2170 FOR K=1 TO 3\L(K)=L(K)-K1\NEXT K
2175 FOR K=1 TO 3\IF L(K)>0 THEN EXIT 2180\NEXT K\GOTO 2220
2180 FOR K=1 TO 3\K2=ASC(N$(J-1+2*K,J-1+2*K))
2185 IF L(K)>0 AND K2>0 AND K2<128 THEN EXIT 2195\NEXT K


2190 GOTO 2220
2195 FILL P1,64+K1\P1=P1+1
2200 FOR K=1 TO 3\IF L(K)<>0 THEN 2210
2205 L(K)=-99\FILL P1,112+32*K+15\P1=P1+1
2210 NEXT K\GOTO 2160
2215 REM LOOK FOR CONSECUTIVE NOTES
2220 FOR K=1 TO 3\IF L(K)>0 OR ASC(N$(J-1+2*K,J-1+2*K))>0 THEN EXIT 2230
2225 NEXT K\GOTO 2285
2230 K3=0\FOR K=1 TO 3
2235 IF N$(J-1+2*K,J-1+2*K)<>N1$(K,K) THEN 2245
2240 IF L(K)=0 AND T1(K)=0 THEN K3=1
2245 NEXT K\IF K3>0 THEN 2260
2250 FILL P1,64+K1\P1=P1+1\IF P1<49000 THEN 2280
2255 PRINT "Song too long."\FILL P1,64\F1=1\EXIT 1010
2260 FILL P1,64+K1-1\P1=P1+1
2265 FOR K=1 TO 3\IF L(K)<>0 OR T1(K)=1 THEN 2275
2270 IF N$(J-1+2*K,J-1+2*K)<>N1$(K,K) THEN 2275
2272 L(K)=-99\FILL P1,112+32*K+15\P1=P1+1
2275 NEXT K\FILL P1,65\P1=P1+1\IF P1>49000 THEN 2255
2280 NEXT J
2285 FILL P1,64+K1\P1=P1+1\FILL P1,64\F1=1
2290 K4=CALL(P2,S)\FILL N,159\FILL N,191
2295 FILL N,223\FILL N,255\GOTO 1010

3000 REM  *** REPLAY A SONG ALREADY TRANSLATED ***
3015 IF F1=1 THEN 3030
3020 PRINT "You must `PLAY´ the song before you `REPLAY´ it."
3025 GOTO 1010
3030 J=CALL(P2,S)\FILL N,159\FILL N,191\FILL N,223\FILL N,255
3035 GOTO 1010

4000 REM  *** SET THE TEMPO ***
4015 PRINT "Old tempo (quarter notes per minute) was",5000/S
4020 INPUT "Enter new tempo:  ",J
4025 IF J>50 AND J<500 THEN 4035
4030 PRINT "That s not realistic!  ",\GOTO 4020
4035 S=INT(5000/J+.5)\GOTO 1010

5000 REM  *** PUT NEW ADDRESS INTO P ***
5015 PRINT "Program Counter presently points to",P
5020 INPUT "Enter new value:  ",J
5025 IF J>=1 AND J<1000 THEN 5035
5030 PRINT "That´s out of bounds.  ",\GOTO 5020
5035 P=INT(J)\GOTO 1010

6000 REM  *** INPUT NOTES TO MEMORY ***
6015 GOSUB 6200
6020 IF F4=1 THEN 1010
6025 FOR J=1 TO 6\N$(6*P-6+J,6*P-6+J)=CHR$(W(J))\NEXT J
6030 P=P+1\F2=0
6035 IF P<1000 THEN 6015
6040 P=P-1
6045 PRINT "End of memory reached."\GOTO 1010
6195 REM  *** SUBROUTINE TO READ 3 NOTES INTO `W´ ARRAY ***
6200 F4=0
6205 PRINT CHR$(12),"Location:  ",P
6210 FOR J=1 TO 3
6215 PRINT "Note",J,
6220 INPUT ":  ",I$
6225 IF LEN(I$)>0 THEN 6235
6230 W(2*J-1)=0\W(2*J)=0\GOTO 6505
6235 IF I$<>"END" THEN 6245
6240 F4=1\EXIT 6510
6245 IF I$(1,1)<>"T" THEN 6285
6250 IF J=1 THEN 6260
6255 PRINT "Tempo can only be changed during `Note 1 ."\GOTO 6215
6260 INPUT "Enter tempo:  ",K
6265 IF K>50 AND K<500 THEN 6275
6270 PRINT "That s not realistic!  ",\GOTO 6260
6275 W(2)=INT(5000/K+.5)\W(1)=255\W(3)=0\W(4)=0\W(5)=0\W(6)=0
6280 EXIT 6510
6285 IF I$(1,1)<>"V" THEN 6315
6290 INPUT "Enter attenuation in dB (0 thru 30):  ",K
6295 IF K>=0 AND K<=30 THEN 6310
6300 PRINT "The attenuation must be between 0dB (full volume)",
6305 PRINT " and 30dB (off)."\GOTO 6290
6310 W(2*J)=INT(K/2+.5)\W(2*J-1)=128\GOTO 6505
6315 IF I$(1,1)<>"R" THEN 6325
6320 W(2*J-1)=48\GOTO 6440
6325 IF LEN(I$)=3 THEN 6345
6330 I$=I$+"N"
6335 IF LEN(I$)=3 THEN 6345
6340 PRINT "Invalid input.  Retype ",\GOTO 6215
6345 K1=0\K2=0\REM  K1=OCTAVE, K2=NOTE
6350 IF I$(1,1)="1" THEN K1=1
```

More

Listing 2 continued.

```
6355 IF I$(1,1)="2" THEN K1=2
6360 IF I$(1,1)="3" THEN K1=3
6365 IF I$(1,1)="4" THEN K1=4
6370 IF I$(1,1)="5" THEN K1=5


6375 IF I$(2,2)="C" THEN K2=2
6380 IF I$(2,2)="D" THEN K2=4
6385 IF I$(2,2)="E" THEN K2=6
6390 IF I$(2,2)="F" THEN K2=7
6395 IF I$(2,2)="G" THEN K2=9
6400 IF I$(2,2)="A" THEN K2=11
6405 IF I$(2,2)="B" THEN K2=13
6410 IF K1=0 OR K2=0 THEN 6340
6415 IF I$(3,3)="S" THEN K2=K2+1
5420 IF I$(3,3)="#" THEN K2=K2+1
6425 IF I$(3,3)="B" THEN K2=K2-1
6430 IF I$(3,3)="b" THEN K2=K2-1
6435 W(2*J-1)=K1*16+K2
6440 REM  GET DURATION
6445 PRINT "Duration",J,
6450 INPUT ":  ",K
6451 IF K<0 THEN K5=128 ELSE K5=0
6452 K=ABS(K)
6455 K1=INT(K)\K2=K-K1
6460 K=48/K1
6465 IF K2=.5 THEN K=K*1.5
6470 IF K2=.3 THEN K=K*2/3
6475 K=INT(K)
6480 IF K>=1 AND K<=48 THEN 6495
6485 PRINT "That's outside the range of permissable values."
6490 PRINT "(Whole note thru sixteenth note) ",\GOTO 6445
6495 W(2*J)=K+K5
6505 NEXT J
6510 RETURN


7000 REM  *** INSERT NOTES TO MEMORY (PUSH FOLLOWING LOCATIONS UP) ***
7015 GOSUB 6200
7020 IF F4=1 THEN 1010
7025 K1=P-INT(P/10)*10
7030 FOR J=990+K1 TO P STEP -10
7035 B$=N$(J*6-5,J*6+54)
7040 N$(J*6+1,J*6+60)=B$
7045 NEXT J
7050 FOR J=1 TO 6\N$(6*P-6+J,6*P-6+J)=CHR$(W(J))
7052 N$(5994+J,5994+J)=CHR$(0)\NEXT J
7055 P=P+1\F2=0
7060 IF P<1000 THEN 7015
7065 P=P-1\PRINT "End of memory reached."\GOTO 1010


8000 REM  *** DELETE NOTES BEGINNING AT `P' ***
8015 PRINT "How many notes should be deleted, starting with note",P,
8020 INPUT ":  ",K
8025 IF P+K<=999 THEN 8035
8030 PRINT "That's too many.  There are only 999 notes."\GOTO 8015
8035 FOR J=P+K TO 999 STEP 10
8040 B$=N$(J*6-5,J*6+54)
8045 N$((J-K)*6-5,(J-K)*6+54)=B$
8050 NEXT J
8055 FOR J=5995-6*K TO 5994\N$(J,J)=CHR$(0)\NEXT J
8060 F2=0\GOTO 1010


9000 REM  *** CLEAR THE NOTE STORAGE ARRAY ***
9015 IF F2=1 THEN 9035
9020 PRINT "Are you sure you want to clear the memory (you haven't saved"
9025 INPUT "the present file since you last changed it)?  ",I$
9030 IF I$<>"YES" THEN 1010
9035 B$=""\FOR J=1 TO 60\B$=B$+CHR$(0)\NEXT J
9040 FOR J=60 TO 6060 STEP 60\N$(J-59,J)=B$\NEXT J
9045 P=1\GOTO 1010


10000 REM  *** SAVE SONG ON DISK ***
10015 INPUT "Which file would you like to use?  ",F$
10017 IF LEN(F$)=0 THEN 1010
10018 IF F$(1,1)<>"F" THEN 10020
10019 PRINT "Sorry, names cannot begin with `F'."\GOTO 1010
10020 K1=FILE(F$)
10025 IF K1>=0 THEN 10050
10030 INPUT "That file doesn't exist.  Shall I create it?  ",I$
10035 IF I$<>"YES" THEN 1010
10040 GOSUB 10300
10045 CREATE F$,K2\GOTO 10065
10050 IF K1=3 THEN 10060
10055 PRINT "Sorry, that's not a BASIC data file."\GOTO 1010
10060 GOSUB 10300
10065 OPEN #1,F$,K1
10070 IF K1>=K2 THEN 10080
10075 PRINT "Sorry, that file isn't large enough."\CLOSE #1\GOTO 1010
10080 WRITE #1,K3
10085 FOR J=1 TO K3\WRITE #1,N$(J*60-59,J*60)\NEXT J
10090 CLOSE #1
10095 F2=1\GOTO 1010
10300 REM  *** SUBROUTINE TO COMPUTE FILE SIZE REQUIRED
10305 REM  K2=# SECTORS, K3=# 10 NOTE BLOCKS
10310 FOR K3=1 TO 100
10315 IF N$(60*K3-5,60*K3-5)<>CHR$(0) THEN 10330
10320 IF N$(60*K3-3,60*K3-3)<>CHR$(0) THEN 10330
10325 IF N$(60*K3-1,60*K3-1)=CHR$(0) THEN EXIT 10335
10330 NEXT K3
10335 K2=INT((K3*62+6)/256)+1
10340 RETURN


11000 REM  *** LOAD A SONG FROM DISK ***
11015 IF F2=1 THEN 11035
```

More

music generation that the notes do not stay synchronized in the short term. Over a long sequence, such as a measure, they do, but often a half note is combined with two quarter notes or a quarter note triplet. This means that as you type the music in, one location may contain a half note for tone generator 1 and a quarter note for tone generator 2. Then the following location will contain nothing for tone generator 1 and a different quarter note for tone generator 2.

This is because the half note being sounded by tone generator 1 is only 50 percent through when tone generator 2 comes to the end of its first quarter note and needs to look for another. In a different circumstance, a half note may be paired with a single quarter note, implying that the second tone generator should remain silent for the second half of the half note. The translating routine must be able to handle these situations.

Translating the Notes

Lines 2000–2295 contain the code that translates the notes and pokes the results into the data area used by the machine-language routine. This section is split into three major blocks.

The first block, lines 2000–2155, fetches a six-byte note triplet from the large string, converts each of the three frequencies to 76489A codes and pokes them to the machine-language routine data table. The frequency byte is used to access the "bottom octave" table via the four bits representing the desired note. Then the value from the table is divided by two the same number of times as the octave specified.

Remember—the 76489A requires a period as an input, not a frequency. Thus, dividing the period by two raises the frequency one octave. The duration of each note is saved for later analysis. If the code for a given tone generator is blank and the previous duration on this generator has expired, it is turned off. If this generator is in the middle of a note, the blank is ignored. If any durations are negative, a flag is set to indicate that this note is to be tied to the next one for this tone generator.

Block number two, line 2160–2210, finds the tone with the shortest duration. This value is subtracted from all three durations. If the next note triplet only contains values for notes that have timed out, the program jumps to block three. If the next note triplet

contains a value for a note that has not timed out yet, the value of the shortest duration is poked to the output table as a delay, and codes to turn off any tones that have timed out are also poked to the output table. Then block two is run again.

The third block, lines 2215–2295, checks for consecutive notes of the same frequency. If any note that has just ended is the same frequency as the corresponding note in the next triplet, a pause equal to a 1/48 note is inserted into the output table. However, if the previous duration was negative, indicating a tied note, the pause is not inserted. After this, return to block 1 to go around again on the next set of notes.

When a couple of all-blank-note triplets are encountered, the play translation routine assumes it has come to the end of the song and pokes a return code to the output table. The machine-language program to play the music is then called, passing the tempo to it as the only parameter. When the song has been played, the machine-language program returns to the BASIC program, which jumps back to the command input section at line 1000.

Replaying a Translated Song

The replay section comprises lines 3000–3035. After checking that the song has been translated, you call the machine-language routine. If the song has not yet been translated, an error message is printed and the command mode reentered.

Bookkeeping Routines

Lines 4000–4035 set the tempo. Since the tempo is usually given in quarter notes per minute, it is converted to the length of a 1/48 note and saved. The tempo can also be stored as part of a song, in which case playing the song will use the pre-stored tempo. However, you can change the tempo after the song is finished and replay it at the new tempo. The entire song must be played at the same tempo; you can't alter the tempo in the middle of a song. This would be a nice improvement for the future.

Lines 5000–5035 load a new address into the pointer used by the editor. Any inserting, inputting or deleting takes place at this point.

Inputting data. Lines 6000–6510 form the input routine. Most of the work is done in a subroutine, lines 6195–6510. This subroutine is also

Listing 2 continued.

```
11020 PRINT "Are you sure you want to load a file (you haven t saved"
11025 INPUT "the present file since you last changed it)?  ",I$
11030 IF I$<>"YES" THEN 1010
11035 INPUT "Which file would you like to load?  ",F$
11037 IF LEN(F$)=0 THEN 1010
11038 IF F$(1,1)<>"F" THEN 11040
11039 PRINT "Sorry, names cannot begin with `F'."\GOTO 1010
11040 K1=FILE(F$)\IF K1>=0 THEN 11050
11045 PRINT "Sorry, that file doesn't exist."\GOTO 1010
11050 IF K1=3 THEN 11060
11055 PRINT "Sorry, that's not a BASIC data file."\GOTO 1010
11060 OPEN #1,F$
11065 READ #1,K3
11070 FOR J=1 TO K3\READ #1,N$(J*60-59,J*60)\NEXT J
11075 CLOSE #1
11080 F1=0\F2=1\P=1\GOTO 1010


12000 REM  *** LIST THE SONG ***
12015 INPUT "Memory range to list (first, last):  ",K1,K2
12020 IF K1<=K2 THEN 12030
12025 PRINT "The first number must be the smaller."\GOTO 12015
12030 IF K1>=1 AND K2<=999 THEN 12040
12035 PRINT "Memory only goes from 1 to 999."\GOTO 12015
12040 INPUT "Would you like `sharp' or `flat' notation?  ",I$
12041 IF LEN(I$)=0 THEN 12055
12043 C3=0\IF I$(1,1)="S" THEN C3=1
12045 IF I$(1,1)="F" THEN C3=2
12050 IF C3>0 THEN 12060
12055 PRINT "Please enter `SHARP' or `FLAT'."\GOTO 12040
12060 INPUT "Would you like the listing printed?  ",I$
12065 F3=0\IF LEN(I$)=0 THEN 12075
12070 IF I$(1,1)="Y" THEN F3=1
12075 FOR J=K1 TO K2
12080 PRINT #F3,J,
12085 FOR K=1 TO 6\W(K)=ASC(N$(J*6-6+K,J*6-6+K))\NEXT K
12090 FOR K=1 TO 3
12095 IF W(K*2-1)=0 THEN 12235
12100 IF W(K*2-1)<>255 THEN 12110
12105 PRINT #F3,TAB(15),"TEMPO  ",5000/W(K*2),\EXIT 12240
12110 IF W(K*2-1)<>128 THEN 12120
12115 PRINT #F3,TAB(K*15),"VOLUME ",2*W(2*K),\GOTO 12235
12120 IF W(K*2-1)<>48 THEN 12130
12125 PRINT #F3,TAB(K*15),"R",\GOTO 12145
12130 K3=INT(W(K*2-1)/16)\K4=W(K*2-1)-K3*16
12135 IF C3=1 THEN PRINT #F3,TAB(K*15),K3,C1$(K4*2-1,K4*2),
12140 IF C3=2 THEN PRINT #F3,TAB(K*15),K3,C2$(K4*2-1,K4*2),
12145 K3=0\K4=W(K*2)
12150 IF K4>127 THEN K4=K4-128
12155 IF K4=48 THEN K3=1
12160 IF K4=32 THEN K3=1.3
12165 IF K4=36 THEN K3=2.5
12170 IF K4=24 THEN K3=2
12175 IF K4=16 THEN K3=2.3
12180 IF K4=18 THEN K3=4.5
12185 IF K4=12 THEN K3=4
12190 IF K4=8 THEN K3=4.3
12195 IF K4=9 THEN K3=8.5
12200 IF K4=6 THEN K3=8
12205 IF K4=4 THEN K3=8.3
12210 IF K4=3 THEN K3=16
12215 IF K4=2 THEN K3=16.3
12216 IF W(K*2)>127 THEN K3=-K3
12220 IF K3<>0 THEN 12230
12225 PRINT #F3,TAB(K*15+7),"INVALID",\GOTO 12235
12230 PRINT #F3,TAB(K*15+7),K3,
12235 NEXT K
12240 PRINT #F3\NEXT J
12245 GOTO 1010


13000 REM  *** SAVE TRANSLATED SONG ON DISK ***
13005 IF F1=1 THEN 13015
13010 PRINT "You must `PLAY' the song before doing a fast-save."\GOTO 1010
13015 INPUT "Which file would you like to use?  ",F$
13020 IF LEN(F$)=0 THEN 1010
13025 IF F$(1,1)="F" THEN 13035
13030 PRINT "Sorry, names of translated songs must begin with `F'."\GOTO 1010
13035 K1=FILE(F$)\IF K1>=0 THEN 13065
13045 INPUT "That file doesn't exist.  Shall I create it?  ",I$
13050 IF I$<>"YES" THEN 1010
13055 K2=INT((P1-41020+12)/256)+1
13060 CREATE F$,K2\GOTO 13080
13065 IF K1=3 THEN 13075
13070 PRINT "Sorry, that's not a BASIC data file."\GOTO 1010
13075 K2=INT((P1-41020+12)/256)+1
13080 OPEN #1,F$,K1
13085 IF K1>=K2 THEN 13095
13090 PRINT "Sorry, that file isn't large enough."\CLOSE #1\GOTO 1010
13095 WRITE #1,S,P1
13100 FOR J=41020 TO P1\WRITE #1,&EXAM(J)\NEXT J
13105 CLOSE #1\GOTO 1010


14000 REM  *** LOAD TRANSLATED SONG FROM DISK ***
14015 INPUT "Which file would you like to load?  ",F$
14020 IF LEN(F$)=0 THEN 1010
14025 IF F$(1,1)="F" THEN 14035
14030 PRINT "Sorry, names of translated songs begin with `F'."\GOTO 1010
14035 K1=FILE(F$)\IF K1>=0 THEN 14045
14040 PRINT "Sorry, that file doesn't exist."\GOTO 1010
14045 IF K1=3 THEN 14055
14050 PRINT "Sorry, that's not a BASIC data file."\GOTO 1010
14055 OPEN #1,F$
14060 READ #1,S,P1
14065 FOR J=41020 TO P1\READ #1,&K2\FILL J,K2\NEXT J
14070 CLOSE #1\F1=1\GOTO 1010
```

used by the insert routine. The input routine just steps through the large note array, filling it with note triplets entered from the keyboard.

The subroutine used to enter notes begins by reading in the octave and note. As notes are entered into the string array, they are encoded to save storage and compile time. Working on one note at a time produces a delay that is not noticeable while entering music and gets part of the translating done early. Each note is entered in the form octave-note-modifier (e.g., 2A, 4F#, 3C or 1Bb).

As soon as it is entered, the note is broken down into the three parts. The octave runs from 1 to 5, with the octave beginning with middle C being 3. This takes three bits. The note (A through G) is converted into a pointer to a table of values, which will produce the desired frequency when loaded into the 76489A.

Finally, the modifier (# or b) is used to offset the pointer by one to point to values corresponding to the black keys on a piano. (For ease in typing, S or F can be used in place of # or b, respectively.) This produces a pointer value running from 1 to 14, which requires four bits to store. So far, seven bits in the byte have been used. The last bit is used as a flag to indicate if this particular byte is not a note but rather a tempo, volume or rest. These each have their own special code.

After processing the note input, the routine asks for the duration of the note. This is entered as a number indicating the size note. A 4 is a quarter note, 16 is a sixteenth note, and so on. The program will handle notes from wholes to sixteenths.

You can add .5 to the duration to indicate a dotted note, or .3 to signify a triplet. These duration values are converted into equivalent 1/48 notes. Thus, 2 becomes 24, 4.5 becomes 18, and 8.3 becomes 4. This number, running from 1 to 63, is stored in the byte immediately following the code for the frequency of the note.

I chose the 1/48 note for the smallest unit because it is the largest number less than 63 that is evenly divisible by 48. Thus, the longest note possible is the whole note (length of 48), and the shortest is the sixteenth note (length of 3). All notes in between can be specified as dotted notes or triplets, or as standard length. If any note is to be tied to the following note, you set the most significant bit of the duration byte to 1; otherwise, it is 0. In this way the code for all three tone generators occupies six bytes of the string.

Inserting notes. Lines 7000–7065 handle the insert routine. This is nearly the same as the input routine, except that before a new triplet is put in the large array the location to be used and all following locations are pushed up one spot. This lets you insert new notes between existing notes.

Deleting notes. Lines 8000–8060 delete notes from the memory. You do this by moving the notes following the undesired group of notes down to lower addresses, thus overwriting the unwanted block of notes.

Clearing memory. Lines 9000–9045 clear the memory by writing blanks to all locations. If the present song has been modified and not been saved, a warning gives you the opportunity to back out if you really did want to save the song first.

Saving song files. Lines 10000–10340 save the present song on the disk. You'll have to rewrite this entire section to match your own mass storage device unless you have the North Star disk system. You use this routine to check for errors and verify that the specified file exists, is the right type and is large enough to hold the song.

Loading files from disk. Lines 11000–11080 load the song memory from the disk. This routine must match your save routine.

Listing a Song

Listing the song in the memory is handled by lines 12000–12245. Since the input routine has partially compressed the data, this section must expand it back out to octave, note and duration before printing. Before this can be done, you must specify whether you prefer sharp or flat notation. Fortunately, most songs are written using mostly sharps or mostly flats, so most notes will be listed in the same form in which they were typed. As for the notes that aren't, you'll just have to get used to them (or rewrite the program).

The list command asks for the range of addresses to be displayed, whether sharp or flat notation is preferred and whether to list to the CRT or printer. Then you begin decoding the compressed data.

First, the frequency byte is decoded. The three bits containing the octave are masked off and saved. The four bits indicating the pointer to the bottom octave table are used to calculate the address of the substring that contains the mnemonics for the notes (such as A# or C). You then print the octave, followed immediately by the note mnemonic.

Since the notes are compressed before they are written into the memory, at this point you don't know whether a particular black key was entered as a sharp or a flat. For instance, both F sharp and G flat are the same frequency, and both indicate the same key on a piano. This is why you must tell the program whether you prefer sharps or flats before the listing begins.

Two strings actually contain the alphabetic representations of the notes: one for flat and the other for sharp. After you print the octave and note, you pull the duration from the memory. It is converted back to the input format, using a series of IF statements since each value can only indicate one possible note size. If you can't find the value after checking all the legal sizes, an error message is printed for the duration.

Saving a Translated Song

Lines 13000–13105 save a translated song on the disk. Again, you'll have to change this section of the program to match your mass storage device. I added this and the next section about a week after the previous sections were written and debugged. Even on a 4 MHz Z-80, the rate of translation of a song is only about 100 note triplets per minute. Thus, songs of 550 notes take 5-1/2 minutes to translate every time they are loaded in and played.

This gets tedious after a few days. This routine peeks at the bytes after they have been written to the output table and saves them on the disk. This way you can reconstruct the output table directy without translating the song each time it is loaded and played, thus saving much time.

Loading a Translated Song

Lines 14000–14070 load a translated song into memory from the disk. This must match the previous section's format.

For those of you who want to dig into the program, Table 1 lists most of the variables used in EDIT2 (Listing 2). Each variable is followed by a brief description of what it represents.

Once you've gotten the two programs loaded into your computer, pick one of your favorite songs and key it in. If it's been many years since

N$(6060)	1010, six-byte note triplets.
B$(60)	A ten-note buffer for use in inserts, deletes and clears.
I$(6)	Holds user's input.
P	Address pointer to N$, between 1 and 999 inclusive.
T(14)	Values for bottom octave frequencies.
V(3)	Volumes of tones 1, 2 and 3, respectively, in dB/2 of attentuation.
O(3)	Corresponding tone generator is on if 1, off if 0.
S	Length of 48th note in milliseconds.
L(3)	Length of time in 48th notes until corresponding tone generator's note is over. If −99, tone generator is off.
J,J1, . . .	Miscellaneous temporary variables.
K,K1, . . .	Miscellaneous temporary variables.
F$(10)	File name for disk access.
C1$(28)	Note names using sharps.
C2$(28)	Note names using flats.
C3	Flag: 1 for sharps, 2 for flats.
N	Address of 76489A.
F1	Flag: 0 for song not played, 1 for song played.
F2	Flag: 0 for file not saved since last change, 1 for file saved since last change.
F3	Flag: 0 for list on CRT, 2 for list on printer.
F4	Flag: 1 for END input during note input, 0 for other input.
W(6)	Working storage for bytes of note triplet.
P1	Pointer to memory for translated bytes.
P2	Machine-language program calling address.
N1$(3)	Note currently being played on corresponding tone generator.
T1(3)	Tie flag: 0 for not tied, 1 for tied.

Table 1. Important variables used in EDIT2 with a description of what they each represent.

you've even looked at a piece of sheet music, reacquaint yourself with how to read music. Fast-paced songs generally come out sounding better than slow ones, so don't start out with "Auld Lang Syne."

User's Guide

After you load the program and type RUN, the initialization routine clears out all the note memory, turns off the 76489A and waits for you to tell it what else to do. To begin entering a song, type RUN. The screen will be cleared and the address to be filled next will be displayed. Initially it will be the first location.

Following the prompts, key in the notes and durations for one, two or three tone generators as required. The note is entered in the format octave-note-modifier. The octave is a number from 1 to 5 inclusive. C is the lowest note in each octave, and middle C is in octave 3. The note is the letter name of the note, from A to G. The modifier indicates whether the note is sharp or flat. # or S indicates sharp, while b or F indicates flat. The modifier is left off for regular notes.

You'll have to remember the key signature of the music you're entering and type each note with the proper modifier. Finally, hit carriage return after you have specified the note as above. If one or more of the three tone generators is not used on a particular note triplet, simply hit carriage return without typing in any note.

After each frequency is typed in and processed, the program will prompt you for the duration. The duration is entered as 1 for a whole note, 2 for a half note, 4 for a quarter note, 8 for an eighth note and 16 for a sixteenth note.

You indicate a dotted note, which is sounded for half again as long as an undotted note, by appending .5 to the length of the note. A musical triplet, as opposed to one of the program's note triplets, refers to a note that is sounded for two-thirds as long as a standard note. It is indicated by appending .3 to the length of the note. Thus, a dotted quarter note is typed in as 4.5, and one note of an eighth note triplet is typed in as 8.3.

To set the tempo as part of the song, type TEMPO when the program asks for the first note of a triplet. To set the volume of a tone generator, type VOLUME when the program prompts for the note. In the above cases the program will prompt you for the necessary information. A rest is indicated by typing R for the note. Don't forget that rests must have a length specified.

Remember that all the tone generators are turned off when the program is loaded and RUN, so if you want to hear the song you're keying in, you'd better set the tone generators to the appropriate volume before entering any of the notes of the song. Otherwise, it will be played in total silence.

When you wish to stop entering notes into the memory, type END for any note in a triplet, and you'll be back in the command mode. The entire triplet containing the END is ignored.

To rework a section of the song already entered, or to continue on after having stopped in the middle of a song the previous day, type GOTO. The program will ask which location you wish to set the pointer to.

To insert notes between existing notes, use the GOTO command to set the pointer to the address you wish to insert ahead of. Then type INSERT and key in notes as described in the input section above. Use END to leave the insert mode.

To delete a note or block of notes from memory, use the GOTO command to position the pointer at the first location to be deleted. Then type DELETE and the program will ask how many locations are to be deleted. This clears the locations to be deleted and pulls down the notes following the last deleted location to fill the gap.

If you change a note, you don't have to delete it and insert the new note. The input function will overwrite the old note with the new one. This also applies if a block of notes is to be modified, as long as you have the same number of notes when you finish as when you began.

To alter the tempo, type TEMPO. The program will tell you what the tempo is set to and let you change it. Playing a song may change the tempo if it is specified as part of the song. That is treated exactly as if it were entered from the keyboard. If the song specifies the tempo, you can still play it at a new tempo by first playing it, then changing the tempo and replaying it at the new tempo.

If you want to start over, type CLEAR. This sets the memory back to blanks and the pointer back to location 1.

To check notes in the memory or document songs, use the list function. After you type LIST, the program will ask the range of locations to list. If you are only interested in one address, enter it for both the first location and the last location. After accepting the address range, the program asks whether you prefer sharp or flat notation. Finally, the program

asks whether you'd like the listing printed on a hard-copy device (if you have one). Type YES for a printed listing, NO or just carriage return for a listing on the CRT.

Once the listing begins, you can't stop it, so if you have specified a large range of memory, it may take some time. The listing does not stop when the end of the song is reached. You need to have some idea where the end is before you start so you can put it in for the last location to be listed.

Once you have typed a song into the memory, the best thing to do is save it on your mass storage device. This is especially important the first few times after you have keyed in the program, when the possibility of bugs is high. It is frustrating to spend a half hour (or more) typing in a song only to get a syntax error when you try to play it. Since restarting the program clears the variables, the song must be reentered.

To save a song, type SAVE. The program will ask for the name of the file to be used. You can use an existing file, as when you are putting the finishing touches on a previously entered song, or use a new file. In the latter case the program will create the file for you. File names cannot begin with the letter F, which is reserved to indicate a file containing a song in translated format (F stands for fast load).

Conversely, type LOAD to read a song into the memory from mass storage. Again, the program asks for the name of the file.

Making Music

Once you have the song typed in and saved, the moment of truth arrives. Type PLAY to translate the song and play it on the 76489A circuitry. Translating takes about one minute per 100 note triplets on a 4 MHz Z-80; estimate your time accordingly.

The first time or two it's good to play the song after the first dozen notes have been entered just to verify that everything is working OK. When it's finished playing, you can go back to enter more simply by typing INPUT; you'll be back where you ended.

To hear the song again without having to re-translate it, type REPLAY, and it will immediately begin playing. At this point you can change the tempo and replay the song at a faster or slower rate.

The commands FSAVE and FLOAD are identical to SAVE and LOAD, except that they are used to save the translated version of a song. The file name must begin with F, and the song must have been played so that a translated version exists.

Listing 3 is J.S. Bach's "Jesu, Joy of Man's Desiring," printed by EDIT2. If you don't recognize the title, you'll probably recognize the song. It uses all three tone generators and is a good demonstration piece. The playing time is about two minutes. Even if you can't read music, you'll at least have one song in your repertoire.

If you have questions, please enclose a stamped, self-addressed envelope. Also, contact me if you want to swap songs on North Star single-density disk systems.■

Listing 3. "Jesu, Joy of Man's Desiring" as listed by EDIT2. This is what the program types for a listing of a song in the memory. Column 1 is the address, columns 2 and 3 are the note and duration of tone generator number 1, columns 4 and 5 are the same for generator number 2, and columns 6 and 7, the same for generator number 3. EDIT2 does not utilize the noise generator.

1	TEMPO	119.04762				
2	VOLUME	4	VOLUME	4	VOLUME	4
3	3B	2	3G	2	1G	4
4					2G	4
5	4C	4	3G	4	2E	4
6	4D	2	3G	2	1B	4
7					1E	4
8	4D	4	3G	4	2B	4
9	3B	2	3G	2	1G	4
10					2G	8
11	4C	4	3G	4	2E	4
12	4D	2.5	3G	2.5	2B	2.5
13	3B	2	3G	2	1G	4
14					2G	4
15	4C	4	3G	4	2E	4
16	4D	2	3G	2	1B	4
17					1E	4
18	4D	4	3G	4	2B	4
19	4C	4	3G	2	2E	2
20	3B	4				
21	3A	4	3F#	4	R	4
22	VOLUME	6	VOLUME	6	VOLUME	6
23	R	8.3			1G	4
24	3G	8.3				
25	3A	8.3				
26	3B	8.3	3D	4.3		
27	4D	8.3				
28	4C	8.3	2F#	8.3		
29	4C	8.3	2G	4.3	2E	4
30	4E	8.3				
31	4D	8.3	2A	8.3		
32	4D	8.3	2G	4	1B	4
33	4G	8.3				
34	4F#	8.3				
35	4G	8.3	2B	4	2E	4
36	4D	8.3				
37	3B	8.3				
38	3G	8.3	1E	4		
39	3A	8.3				
40	3B	8.3				
41	4C	8.3	2A	4.3	1A	4
42	4D	8.3				
43	4E	8.3	2F#	8.3		
44	4D	8.3	2G	4.3	1B	4
45	4C	8.3				
46	3B	8.3	2D	8.3		
47	3A	8.3	2E	4.3	2C	4
48	3B	8.3				
49	3G	8.3	2B	8.3		
50	3F#	8.3	2A	4	2D	4
51	3G	8.3				
52	3A	8.3				
53	3D	8.3	2A	4	2F#	4
54	3F#	8.3				
55	3A	8.3				
56	4C	8.3	2A	4	2D	4
57	3B	8.3				
58	3A	8.3				
59	3B	8.3	2G	4		
60	3G	8.3				
61	3A	8.3				
62	3B	8.3	2G	4.3	2E	4
63	4D	8.3				
64	4C	8.3	2F#	8.3		
65	4C	8.3	2G	4.3	2C	4
66	4E	8.3				
67	4D	8.3	2A	8.3		
68	4D	8.3	2B	4	1B	4
69	4G	8.3				
70	4F#	8.3				
71	4G	8.3	2B	4	2E	4
72	4D	8.3				
73	3B	8.3				
74	3G	8.3	2B	4	2D	4
75	3A	8.3				
76	3B	8.3				
77	3E	8.3	2A	4.3	2C	4
78	4D	8.3				
79	4C	8.3	2F#	8.3		
80	3B	8.3	2G	4.3	2C#	4
81	3A	8.3				
82	3G	8.3	2E	8.3		
83	3D	8.3	2A	4.3	2D	4
84	3G	8.3				

More

Listing 3 continued.

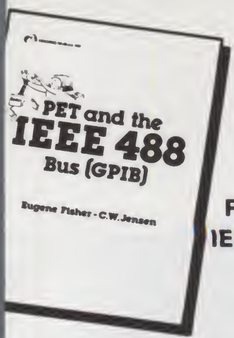
85	3F#	8.3	3C	8.3		
86	3G	8.3	1G	4		
87	3B	8.3				
88	4D	8.3				
89	4G	8.3	1G	4.3		
90	4D	8.3				
91	3B	8.3	2G	8.3		
92	3G	8.3	2D	4.3		
93	3B	8.3				
94	4D	8.3	1B	8.3		
95	VOLUME	6	VOLUME	2	VOLUME	2
96	R	4	3B	2	3G	2
97	R	4				
98	VOLUME	2	VOLUME	2	VOLUME	2
99	4C	4	3G	4	3E	4
100	4D	2	3A	4	2A	2
101			3G	4		
102	4D	4	3F#	4	2B	4
103	4C	2	3G	4	2B	4
104			3D	2	2A	4
105	3B	4			2G	4
106	3A	8.3	2F#	4	2D	4
107	3D	8.3				
108	3E	8.3				
109	3F#	8.3	3D	4.3		
110	3A	8.3				
111	3G	8.3	2D	8.3		
112	3A	8.3	2D	4.3		
113	4C	8.3				
114	3B	8.3	2D	8.3		
115	4C	8.3	1D	-4		
116	3A	8.3				
117	3F#	8.3				
118	3D	8.3	1D	4.3		
119	3F#	8.3				
120	3A	8.3	2D	8.3		
121	4C	8.3	2D	4.3		
122	3B	8.3				
123	3A	8.3	2D	8.3		
124	3B	2	3G	2	3D	2
125	4C	4	3G	4	3E	4
126	4D	2	3G	4	3D	4
127			3D	4	2B	4
128	3B	4	3G	4	2B	4
129	3A	8	3E	4	3C	8
130	3B	16			2A	8
131	4C	16				
132	3B	4	3F#	8	3D	8
133			3E	8	2B	8
134	3A	4	3F#	4	3C	4
135	3G	8.3	3D	4	2B	4
136	3B	8.3				
137	3A	8.3				
138	3B	8.3	3D	4	2G	4
139	4D	8.3				
140	4C	8.3				
141	4C	8.3	2G	4	2E	4
142	4E	8.3				
143	4D	8.3				
144	4D	8.3	2G	4	1B	4
145	4G	8.3				
146	4F#	8.3				
147	4G	8.3	2B	4	2E	4
148	4D	8.3				
149	3B	8.3				
150	3G	8.3	3E	4.3	1E	4
151	3A	8.3				
152	3B	8.3	3D	8.3		
153	4C	8.3	3C	4.3	2E	4
154	4D	8.3				
155	4E	8.3	2F#	8.3		
156	4D	8.3	2G	4	1B	4
157	4C	8.3				
158	3B	8.3				
159	3A	8.3	2E	4.3	2C	4
160	3B	8.3				
161	3G	8.3	2B	8.3		
162	3F#	8.3	2A	4	2D	4
163	3G	8.3				
164	3A	8.3				
165	3D	8.3	2A	4	2F#	4
166	3F#	8.3				
167	3A	8.3				
168	4C	8.3	2D	4		
169	3B	8.3				
170	3A	8.3				
171	3B	8.3	2G	4		
172	3G	8.3				
173	3A	8.3				
174	3B	8.3	2G	4.3	2E	4
175	4D	8.3				
176	4C	8.3	2F#	8.3		
177	4C	8.3	2G	4.3	2C	4
178	4E	8.3				
179	4D	8.3	2A	8.3		
180	4D	8.3	2B	4	1B	4
181	4G	8.3				
182	4F#	8.3				
183	4G	8.3	2B	4	2E	4
184	4D	8.3				
185	3B	8.3				
186	3G	8.3	2B	4	2D	4
187	3A	8.3				
188	3B	8.3				
189	3E	8.3	2A	4.3	2C	4
190	4D	8.3				
191	4C	8.3	2F#	8.3		
192	3B	8.3	2G	4.3	2C#	4
193	3A	8.3				

More

Listing 3 continued.

194	3G	8.3	2E	8.3		
195	3D	8.3	2A	4.3	2D	4
196	3G	8.3				
197	3F#	8.3	3C	8.3		
198	VOLUME	0	VOLUME	0	VOLUME	0
199	3B	2	3G	2	3D	2
200	4C	4	3G	4	3E	4
201	4D	2	3A	4	2A	2
202			3G	4		
203	4D	4	3F#	4	2B	4
204	4C	2	3G	4	2B	4
205			3D	2	2A	4
206	3B	4			2G	4
207	3A	8.3	2F#	4	2D	4
208	3D	8.3				
209	3E	8.3				
210	3F#	8.3	3D	4.3		
211	3A	8.3				
212	3G	8.3	2D	8.3		
213	3A	8.3	2D	4.3		
214	4C	8.3				
215	3B	8.3	2D	8.3		
216	4C	8.3	1D	-4		
217	3A	8.3				
218	3F#	8.3				
219	3D	8.3	1D	4.3		
220	3F#	8.3				
221	3A	8.3	2D	8.3		
222	4C	8.3	3D	4.3	2D	4.3
223	3B	8.3				
224	3A	8.3	3D	8.3	2D	8.3
225	VOLUME	2	VOLUME	2	VOLUME	2
226	3B	2	3G	2	3D	2
227	4C	4	3G	4	3E	4
228	4D	2	3G	4	3D	4
229			3D	4	2B	4
230	3B	4	3E	4	2B	4
231	3A	8	3E	4	3C	4
232	3B	16				
233	4C	16				
234	3B	4	3F#	8	3D	4
235			3E	8		
236	3A	4	3F#	4	3C	4
237	VOLUME	6	VOLUME	6	VOLUME	6
238	3G	8.3	1G	4		
239	3B	8.3				
240	3A	8.3				
241	3B	8.3	3D	4	2G	4
242	4D	8.3				
243	4C	8.3				
244	4C	8.3	2G	4	2E	4
245	4E	8.3				
246	4D	8.3				
247	4D	8.3	2G	4	1B	4
248	4G	8.3				
249	4F#	8.3				
250	4G	8.3	2B	4	2E	4
251	4D	8.3				
252	3B	8.3				
253	3G	8.3	1E	4		
254	3A	8.3				
255	3B	8.3				
256	4C	8.3	2E	4.3	1A	4
257	4D	8.3				
258	4E	8.3	2F#	8.3		
259	4D	8.3	2G	4	1B	4
260	4C	8.3				
261	3B	8.3				
262	3A	8.3	2E	4.3	2C	4
263	3B	8.3				
264	3G	8.3	2B	8.3		
265	3F#	8.3	2A	4	2D	4
266	3G	8.3				
267	3A	8.3				
268	3D	8.3	2A	4	2F#	4
269	3F#	8.3				
270	3A	8.3				
271	4C	8.3	3F#	4	2D	4
272	3B	8.3				
273	3A	8.3				
274	3B	8.3	3D	4	2G	4
275	3G	8.3				
276	3A	8.3				
277	3B	8.3	2G	4.3	2E	4
278	4D	8.3				
279	4C	8.3	2F#	8.3		
280	4C	8.3	2G	4.3	2C	4
281	4E	8.3				
282	4D	8.3	2A	8.3		
283	4D	8.3	2B	4	1B	4
284	4G	8.3				
285	4F#	8.3				
286	4G	8.3	2B	4	2E	4
287	4D	8.3				
288	3B	8.3				
289	3G	8.3	2B	4	2D	4
290	3A	8.3				
291	3B	8.3				
292	3D	8.3	2A	4.3	2C	4
293	4D	8.3				
294	4C	8.3	2F#	8.3		
295	3B	8.3	2G	4.3	2C#	4
296	3A	8.3				
297	3G	8.3	2E	8.3		
298	3D	8.3	2A	4.3	2D	4
299	3G	8.3				
300	3F#	8.3	3C	8.3		
301	3G	8.3	2B	4	1G	8.3
302	3B	8.3			R	8.3

(More →



PET and the IEEE 488 Bus (GPIB)
by E. Fisher and C. W. Jensen

is the only complete guide available on
facing PET to GPIB. Learn how to program
PET interface to control power supplies,
al sources, signal analyzers and other
uments. It's full of practical information, as
of its authors assisted in the original design
e PET GPIB interface.
#31-4 $15.99

NEW PET/CBM edition

Some Common BASIC Programs
by L. Poole, M. Borchers, C. Donahue

rograms you can use even if you don't
v BASIC. This book gives you a variety of
n power including personal finance, taxes
statistics as well as other programs you'll
t like Recipe Cost and Check Writer. All
rams can be run on a PET or CBM with 8K
ore.
#40-3 $14.99

owners can purchase the programs ready-
n on cassette or disk. Use the book as a
ual for operating instructions and
ramming options.
Disk #33-0 $22.50
Cassette #25-X $15.00

ctical BASIC Programs
on Poole

e are 40 easy to use programs that
do something useful.

me averaging, checkbook reconciliation,
atics, factorials, temperature conversion
musical transposition are just a few. It offers
alth of practical computing power. Includes
-ups, program notes and instructional
nples to help you realize the potential uses
ch program.
#38-1 $15.99

02 sembly Language gramming
Leventhal

ase the capabilities and performance of
(and other 6502-based computers) by
ing to program in assembly language.
#27-6 $16.99

✓Reader Service—see page 226

New for your PET

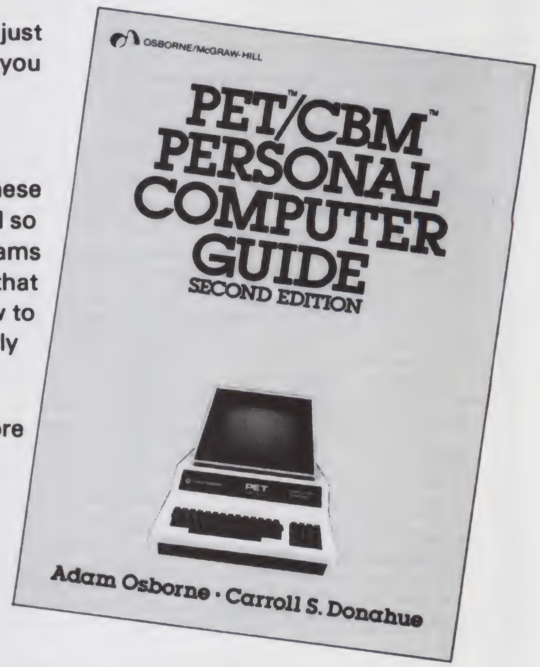
from OSBORNE/McGraw-Hill

PET™/CBM™ Personal Computer Guide Second Edition

by Adam Osborne and Carroll S. Donahue

The PET/CBM Personal Computer Guide is a step-by-step guide that assumes no prior knowledge of computers. If you can read English, you can use this book.

This book provides the important information and documentation that PET/CBM users have sought for so long. After reading this book you will have a good understanding of what a computer — especially the PET/CBM computer — can do for you. If you've just bought a PET or CBM this is the book you must have to really understand your computer. By using the examples found in this book, you will quickly get your PET/CBM up and running. These examples are thoroughly documented so you can learn how and why the programs work. It is the "how" and the "why" that are important if you want to learn how to make your PET or CBM work efficiently for you.



This second edition contains even more useful information than the first edition of this book.
The guide contains a wealth of information on everything from keyboard variations to a detailed description of PET and CBM memory.

Included are:

Complete operating instructions for
- keyboard
- tape cassette
- disk

Description of all CBM BASIC statements
Optimal programming techniques including
- input/output programming
- file handling
- screen editing

Solutions to programming problems
CBM capabilities and limitations

#55-1 $15.00

Book/Cassette/Disk	Price	Quantity	Amount
27-6 6502 Assembly Language Programming	$16.99		
55-1 PET Personal Computer Guide	$15.00		
31-4 PET and the IEEE 488 Bus (GPIB)	$15 99		
40-3 Some Common BASIC Programs PET/CBM ed. (book)	$14.99		
25-X Some Common BASIC Programs PET Cassette	$15 00		
33-0 Some Common BASIC Programs PET Disk	$22 50		
38-1 Practical BASIC Programs	$15.99		
	California resident tax		
	Shipping		
	Total Amount Enclosed		

California residents add 6% sales tax.
S.F. BART residents add 6½% sales tax.

Shipping: (Shipping for large orders to be arranged)
- All foreign orders $4.00 per book for airmail
- $0.45 per book 4th class in the U.S. (allow 3-4 weeks)
- $0.75 per book UPS in the U.S. (allow 10 days)
- $1.50 per book special rush shipment by air in the U.S.

Cassettes and Disk:
- No additional charge in the U.S
- $1.50 each foreign airmail

OSBORNE/McGraw-Hill ✓149
630 Bancroft Way, Dept. K7
Berkeley, California 94710
(415) 548-2805 · TWX 910-366-7277

Name: ____________________

Address: ____________________

City: ____________________

State: __________ ZIP: __________

Phone: ____________________

☐ VISA ☐ Master Card

Credit Card No.: ____________________

1110

Listing 3 continued.

303	4D	8.3			R	8.3
304	4G	8.3	1G	4		
305	4D	8.3				
306	3B	8.3				
307	3G	8.3	3D	4	2F#	4
308	3B	8.3				
309	4C#	8.3				
310	4D	2	3A	2	3F#	2
311	3B	4	3G#	4	3D	4
312	4C	2	3A	2	3E	2
313	4C	4	3A	4	3F	4
314	3B	-4	3A	4	2D	4
315	3B	8	3D	4	1B	4
316	4C	16				
317	4D	16				
318	3B	4	3G	4	2E	4
319	3A	8.3	1A	4		
320	4C	8.3				
321	3B	8.3				
322	4C	8.3	2A	4.3	2F	4
323	4E	8.3				
324	4D	8.3	2G#	8.3		
325	4D	8.3	2A	4.3	2F#	4
326	4F	8.3				
327	4E	8.3	2B	8.3		
328	4E	8.3	3C	4	2C	4
329	4A	8.3				
330	4G#	8.3				
331	4A	8.3	2A	4	2F	4
332	4E	8.3				
333	4C	8.3				
334	3A	8.3	3C	4	2E	4
335	3B	8.3				
336	4C	8.3				
337	4F	8.3	2A	4.3	2D	4
338	4E	8.3				
339	4D	8.3	2G	8.3		
340	4C	8.3	2A	4	2D#	4
341	3B	8.3				
342	3A	8.3				
343	3E	8.3	2B	4.3	2E	4
344	3A	8.3				
345	3G#	8.3	3D	8.3		
346	VOLUME	0	VOLUME	0	VOLUME	0
347	4C	2	3A	2	3E	2
348	4D	4	3G	4	3D	4
349	4E	2	3G	4	3C	4
350			4C	4	2A	4
351	4E	4	3G	4	3C	4
352	4D	-4	3F	4	2A	2
353	4D	8	3A	4		
354	4E	16				
355	4F	16				
356	4D	4	3G	4	2B	4
357	4C	4	3G	4	2C	4
358	4G	8.3	2E	4		
359	4E	8.3				
360	4C	8.3				
361	3G	8.3	3E	4.3	2C	4
362	3A	8.3				
363	3A#	8.3	3D	8.3		
364	3A	8.3	3C	4	2F	4
365	4C	8.3				
366	3B	8.3				
367	4C	8.3	2F	4	2D	4
368	3A	8.3				
369	3F	8.3				
370	3D	8.3	2B	4.3	2G	4
371	3E	8.3				
372	3F	8.3	3D	8.3		
373	3E	8.3	3C	4.3	2C	4
374	3C	8.3				
375	3D	8.3	2G	8.3		
376	3E	8.3	3C	4.3		
377	3G	8.3				
378	3F#	8.3	2C	8.3		
379	3G	8.3	3D	4.3	1B	4.3
380	3B	8.3				
381	3A	8.3	3F#	8.3	2D	8.3
382	3B	2	3G	2	3D	2
383	4C	4	3G	4	3E	4
384	4D	2	3G	2	3D	2
385	4D	4	3G	4		
386	4C	4	3C	4	3G	8
387					3F#	8
388	3B	2	3G	2	3D	4
389					3E	4
390	3A	8.3	2A	4.3	2D	4
391	3D	8.3				
392	3E	8.3	3C#	8.3		
393	3F#	8.3	3D	4.3	2D	4.3
394	3A	8.3				
395	3G	8.3	2D	8.3	1D	8.3
396	3A	8.3	2D	4.3	1D	4.3
397	4C	8.3				
398	3B	8.3	2D	8.3	1D	8.3
399	4C	8.3	2D	-4	1D	-4
400	3A	8.3				
401	3F#	8.3				
402	3D	8.3	2D	4.3	1D	4.3
403	3F#	8.3				
404	3A	8.3	2D	8.3		
405	4C	8.3	2D	4.3		
406	3B	8.3				
407	3A	8.3	2D	8.3		
408	3B	2	3G	2	3D	2
409	4C	4	3G	4	3E	4
410	4D	2	3G	2	3D	2
411	3B	4	3G	4	3E	4
412	3A	8	3E	4	3C	8

More

Listing 3 continued.

413	3B	16			2A	8
414	4C	16				
415	3A	2	3F#	8	3D	8
416			3E	8	2B	8
417			3F#	4	3C	4
418	3G	8.3	R	4	1G	-2.5
419	3B	8.3				
420	4D	8.3				
421	4G	8.3	R	4.3		
422	4D	8.3				
423	3B	8.3	2G	8.3		
424	3G	8.3	2D	4.3		
425	3B	8.3				
426	4D	8.3	2G	8.3		
427	4F	8.3	2B	4.3	1G	-2.5
428	4D	8.3				
429	3B	8.3	R	8.3		
430	3G	8.3	R	4.3		
431	3B	8.3				
432	4D	8.3	2G	8.3		
433	4E	8.3	2E	4.3		
434	4C	8.3				
435	3F	8.3	2C	8.3		
436	3F#	8.3	2A	-4	1G	4
437	3A	8.3				
438	4C	8.3				
439	4D	8.3	2A	4.3	1G	-2
440	3B	8.3				
441	3G	8.3	3D	8.3		
442	3E	8.3	2B	4.3		
443	3G	8.3				
444	3B	8.3	3E	8.3		
445	4C	8.3	2E	4.3	1G	2.5
446	3A	8.3				
447	3F#	8.3	2A	8.3		
448	3D	8.3	2F#	4		
449	3F#	8.3				
450	3A	8.3				
451	4C	8.3	3F#	4.3		
452	3B	8.3				
453	3A	8.3	3D	8.3		
454	3B	8.3	3G	4	1G	4
455	3G	8.3				
456	3A	8.3				
457	3B	8.3	3D	4	2G	4.3
458	4D	8.3				
459	4C	8.3			2F#	8.3
460	4C	8.3	2G	4.3	2E	4
461	4E	8.3				
462	4D	8.3	2A	8.3		
463	4D	8.3	2B	4	1B	4
464	4G	8.3				
465	4F#	8.3				
466	4G	8.3	2B	4	2E	4
467	4D	8.3				
468	3B	8.3				
469	3G	8.3	3E	4.3	1E	4
470	3A	8.3				
471	3B	8.3	3D	8.3		
472	4C	8.3	2A	4.3	1A	4
473	4D	8.3				
474	4E	8.3	2F#	8.3		
475	4D	8.3	2G	4.3	1B	4
476	4C	8.3				
477	3B	8.3	2D	8.3		
478	3A	8.3	2E	4.3	2C	4
479	3B	8.3				
480	3G	8.3	2B	8.3		
481	3F#	8.3	2A	4	2D	4
482	3G	8.3				
483	3A	8.3				
484	3D	8.3	2A	4	2F#	4
485	3F#	8.3				
486	3A	8.3				
487	4C	8.3	2A	4.3	2D	4
488	3B	8.3				
489	3A	8.3	2F#	8.3		
490	3B	8.3	3D	4	2G	4
491	3G	8.3				
492	3A	8.3				
493	3B	8.3	2G	4.3	2E	4
494	4D	8.3				
495	4C	8.3	2F#	8.3		
496	4C	8.3	2G	4.3	2C	4
497	4E	8.3				
498	4D	8.3	2A	8.3		
499	4D	8.3	2B	4	1B	4
500	4G	8.3				
501	4F#	8.3				
502	4G	8.3	2B	4	2E	4
503	4D	8.3				
504	3B	8.3				
505	3G	8.3	2B	4	2D	4
506	3A	8.3				
507	3B	8.3				
508	3E	8.3	2G	4.3	2C	4
509	4D	8.3				
510	4C	8.3	2F#	8.3		
511	3B	8.3	2G	4	2C#	4
512	3A	8.3				
513	3G	8.3				
514	3D	8.3	2A	4.3	2D	4
515	3G	8.3				
516	3F#	8.3	3C	8.3		
517	3G	2.5	3D	2.5	2B	2.5
518						
519						
520						

MATTERS OF CHOICE FOR MANUFACTURING SYSTEMS

Don Ludlum, President and Founder of Ludlum Measurements, Inc. is shown with several of the Radiation Detection Products which are manufactured by the company. Ludlum Measurements, Inc. received the Texas Governor's Industrial Expansion Award in 1975.

THE COMPANY:
LUDLUM MEASUREMENTS, INC.
SWEETWATER, TEXAS

Ludlum Measurements, Inc. was organized in 1962 for the design and manufacture of proprietary radiation detection safety instrumentation. The development and use of radiation sources has moved far beyond X-Ray machines and this new growth has required the development of more sophisticated methods of measuring radiation. From these new applications has developed the need for products which increase the level of nuclear safety. Products were developed by Ludlum Measurements for monitoring nuclear power generating plants, nuclear gauging and servicing applications, nuclear medicine research, energy conservation and research, oil field tubing inspection needs, and many areas of environmental control. Ludlum Measurements has grown from a small sales volume and workforce to a multi-million dollar company employing over 80 people. This growth has created jobs, provided training, and contributed to the overall economic growth of the area. For this contribution the company received the Texas Governor's Industrial Expansion Award in 1975.

"This growth has not been without its problems, particularly in a manufacturing company. The increased sales and labor force meant new challenges for planning and control of the operation," Don Ludlum, Founder and President comments.

The modern production facility at Ludlum Measurements, Inc. employs approximately eighty people. With many different production work orders active at any given time, and many items entering and leaving the stock room, requires the MSI computer system for effective inventory management and control.

"The requirements for increased information obviously lead many companies to consider computerization. The consideration of a computer for a manufacturing company is many times stimulated more by sheer growth than anything else. Coping with increased buying levels, projecting component demands, scheduling production runs, all become larger more complex tasks. We even had a service bureau for some four years, but the costs kept getting higher and we had special problems requiring manufacturing oriented software different than the service bureau could offer."

"We considered IBM and NCR because of name and reputation, but found very quickly that making the right choice was more than a matter of the right name or even price for that matter. We needed an approach that would meet OUR needs. We did a thorough search and we think we made a reasonable judgment of the systems on the market and chose MSI. Based upon the product, software and support, we are glad we chose MSI."

The production of many different models of radiation detection instrumentation requires an inventory of several thousand individual items and subassemblies.

This PET program demonstrates the utility of nested subroutines. You'll have fun with it, too.

Scramble

By Edward Rager

Scramble is a word manipulation game designed to help you at Scrabble, anagrams or scrambled word puzzles. More importantly for programmers, it illustrates how to let nested subroutines take a lot of the work out of programming.

Scramble is designed to run on the Commodore PET, but should be adaptable to any system with string-handling capabilities.

The program takes a word of from three to six letters and displays all possible combinations of these letters on the screen. The number of combinations is found by taking the factorial of the number of letters. For three letters it is the 3 factorial (3!), or $1 \times 2 \times 3 = 6$ (Table 1). For four letters (Table 2) it is 4!: $1 \times 2 \times 3 \times 4 = 24$. Five letters have 120 combinations, and there are 720 possibilities for six letters.

Forty-four words at a time can be displayed on the screen, in four columns of 11 double-spaced rows.

If your only interest is in using this program, then this is the place to stop. Read on, however, if you want to see how laziness can pay off in programming.

How It Works

The word you input for processing is W$. If it is three letters, it is sent to line 3000, where the word is now sent to subroutine 3500 as A3$. Any word sent to subroutine 3500 must go as A3$. This allows any three-letter word to be designated A3$ and sent to subroutine 3500 for processing.

As you will see later, this flexibility lets the program call on subroutine 3500 120 times with just one user input of a six-letter word. Subroutine 3500 divides A3$ into three individual letters (A$, B$ and C$) and puts them back together in all six possible combinations.

If the user inputs a four-letter word, it goes to line 4000, where it is called A4$ and is sent to subroutine 4500. Here the first letter of A4$ is split from the other three. The three are labeled A3$ and sent to subroutine 3500 to be rearranged into six possible combinations. When they return they are sent to subroutine 4900, where they are rejoined with the first letter, which had been held back. Then the second letter of A4$ is held back while the other three are juggled in subroutine 3500. This is repeated until each of the four letters has been kept back and then reunited with the six combinations of the other three, making 24 possible arrangements.

A five-letter word goes to subroutine 5500 as A5$. Four letters of A5$ are sent to subroutine 4500 as A4$, and three letters of A4$ are sent to subroutine 3500 as A3$. These subroutines call on one another until all combinations have been made. To process a five-letter word subroutine 5500 calls subroutine 4500 five times and subroutine 4500 calls subroutine 3500 20 times.

All the subroutines are used to process a six-letter word. Subroutine 3500 is called 120 times. The six-letter words are handled differently from the others, in that not all 720 combinations are saved. It isn't necessary since they don't need to be passed on to a higher subroutine; anyway, an 8K PET would run out of

TAP
TPA
ATP
APT
PTA
PAT

Table 1. Combinations for three letters.

STOP	STPO	SOTP	SOPT
SPTO	SPOT	TSOP	TSPO
TOSP	TOPS	TPSO	TPOS
OSTP	OSPT	OTSP	OTPS
OPST	OPTS	PSTO	PSOT
PTSO	PTOS	POST	POTS

Table 2. Combinations for four letters.

Edward Rager, 9360 Tasmania Ave., Baton Rouge, LA 70810.

Listing 1.

```
SCRAMBLE  BY EDWARD RAGER
 100 DIM A4$(24),A5$(120),A6$(120)
 120 :
 150 OPEN 1,0,1  :  REM  THIS ALLOWS AN
 160 REM  INPUT#1 TO BE USED FOR INPUT
 170 :
 200 PRINT "•SCRAMBLE REARRANGES THE SPE
LLING OF"
 210 PRINT "↓WORDS AND LISTS ALL COMBINA
TIONS OF THE"
 220 PRINT "↓LETTERS. THE WORDS MUST BE
FROM 3 TO 6"
 230 PRINT "↓LETTERS LONG."
 232 PRINT "↓↓TO STOP PROGRAM PREMATUREL
Y TYPE 'STOP'"
 234 PRINT "↓INSTEAD OF 'OK'"
 240 PRINT "↓↓TYPE THE WORD TO BE REARRA
NGED "; : INPUT#1,W$ : PRINT
 248 :
 250 REM  W$ IS THE WORD TO BE JUGGLED
 252 :
 260 L=LEN(W$) : ON L GOTO 300,300,3000,
4000,5000,6000,300
 300 PRINT "↓↓THE WORD MUST BE 3,4,5,OR
6 LETTERS LONG"
 310 GOTO 240
 2800 :
 2820 :
 3000 A3$=W$   :  GOSUB 3500
 3100 PRINT"•"
 3110 :
 3120 FOR I=1 TO 6
 3140 : PRINT A3$(I)
 3150 : PRINT
 3160 NEXT I
 3180 GOTO 8000
 3200 :
 3220 :
 3400 REM  SUB 3500 MAKES ALL POSSIBLE
 3420 REM  REARRANGEMENTS OF 3 LETTER
 3440 REM  WORD A3$.
 3460 :
 3500 A$=LEFT$(A3$,1) : B$=MID$(A3$,2,1)
 3520 C$=RIGHT$(A3$,1):  A3$(1)=A$+B$+C$
 3540 A3$(2)=A$+C$+B$ :  A3$(3)=B$+A$+C$
 3560 A3$(4)=B$+C$+A$ :  A3$(5)=C$+A$+B$
 3580 A3$(6)=C$+B$+A$ :  RETURN
 3900 :
 3920 :
 4000 A4$=W$    :  GOSUB 4500
 4020 PRINT "•" :  T=0
 4100 FOR I=1 TO 24
 4110 : T=T+1
 4120 : PRINT A4$(I),
 4140 : IF T=4 THEN T=0 : PRINT
 4160 NEXT I
 4180 GOTO 8000
 4200 :
 4210 :
 4400 REM  SUB 4500 KEEPS BACK ONE
 4420 REM  LETTER AND SENDS THE OTHER 3
 4440 REM  TO SUB 3500 FOR REARRANGEMENT
 4460 REM  IT DOES THIS FOR EACH OF THE
 4470 REM  4 LETTERS.
 4480 :
 4500 D$=LEFT$(A4$,1) : E$=RIGHT$(A4$,3)
 4520 A3$=E$ : GOSUB 3500
 4540 I1=1 : I2=6 : GOSUB 4900
 4560 D$=MID$(A4$,2,1) : E$=LEFT$(A4$,1)
+RIGHT$(A4$,2)
 4580 A3$=E$ : GOSUB 3500
 4600 I1=7 : I2=12 : GOSUB 4900
 4620 D$=MID$(A4$,3,1) : E$=LEFT$(A4$,2)
+RIGHT$(A4$,1)
 4640 A3$=E$ : GOSUB 3500
 4660 I1=13 : I2=18 : GOSUB 4900
 4700 D$=RIGHT$(A4$,1) : E$=LEFT$(A4$,3)
 4720 A3$=E$ : GOSUB 3500
 4740 I1=19 : I2=24 : GOSUB 4900
 4800 RETURN
 4820 :
 4830 REM  SUB 4900 REJOINS THE HELD-
 4840 REM  BACK LETTER WITH THE OTHER
 4850 REM  3 FROM SUB 3500.
 4860 :
 4900 J=0
 4910 FOR I=I1 TO I2
 4920 : J=J+1
 4940 : A4$(I)=D$+A3$(J)
 4960 NEXT I
 4980 RETURN
 4990 :
 5000 A5$=W$ : GOSUB 5500
 5020 T=0 : II=0 : PRINT "•"
 5050 REM  PRINTOUT WILL LIST 11 LINES
 5060 REM  OF 4 WORDS EACH ON SCREEN
 5100 FOR I=1 TO 120
 5120 : T=T+1 : PRINT A5$(I),
 5140 : IF T=4 THEN T=0 : II=II+1 :PRINT
 5160 : IF II<>11 THEN 5180
 5170 : PRINT "TYPE  OK  TO CONTINUE ",
: INPUT#1,B$
 5175 : GOSUB 9000 : PRINT :´T=0 : II=0
 5177 : PRINT "•"
 5180 NEXT I
 5200 GOTO 8000
 5220 :
 5230 :
 5400 REM  SUB 5500 KEEPS BACK ONE
 5420 REM  LETTER AND SENDS THE OTHER 4
 5440 REM  TO SUB 4500 FOR REARRANGEMENT
 5460 REM  IT DOES THIS FOR EACH OF THE
 5470 REM  5 LETTERS.
 5480 :
 5500 G$=LEFT$(A5$,1) : H$=RIGHT$(A5$,4)
 5510 A4$=H$:GOSUB 4500 : I1=1:I2=24:GOS
UB 5900
 5530 G$=MID$(A5$,2,1) : H$=LEFT$(A5$,1)
+RIGHT$(A5$,3)
 5540 A4$=H$ : GOSUB 4500 : I1=25 : I2=4
8 : GOSUB 5900
 5560 G$=MID$(A5$,3,1) : H$=LEFT$(A5$,2)
+RIGHT$(A5$,2)
 5570 A4$=H$ : GOSUB 4500
 5580 I1=49 : I2=72 : GOSUB 5900
 5590 G$=MID$(A5$,4,1) : H$=LEFT$(A5$,3)
+RIGHT$(A5$,1)
 5600 A4$=H$ : GOSUB 4500
 5610 I1=73 : I2=96 : GOSUB 5900
 5620 G$=RIGHT$(A5$,1) : H$=LEFT$(A5$,4)
 5630 A4$=H$ : GOSUB 4500
 5640 I1=97 : I2=120 : GOSUB 5900
 5650 RETURN
 5660 :
 5830 REM  SUB 5900 REJOINS THE HELD-
 5840 REM  BACK LETTER WITH THE OTHER
 5850 REM  4 FROM SUB 4500.
 5860 :
 5900 J=0
 5910 FOR I=I1 TO I2
 5920 : J=J+1 : A5$(I)=G$+A4$(J)
 5940 NEXT I
 5960 RETURN
 5970 :
 6000 A6$=W$ : GOSUB 6500 : GOTO 8000
 6100 :
 6120 :
 6400 REM  SUB 6500 KEEPS BACK ONE
 6420 REM  LETTER AND SENDS THE OTHER 5
 6440 REM  TO SUB 5500 FOR REARRANGEMENT
 6460 REM  IT DOES THIS FOR EACH OF THE
 6470 REM  6 LETTERS.
 6480 :
 6500 M$=LEFT$(A6$,1) : N$=RIGHT$(A6$,5)
 6510 A5$=N$ : GOSUB 5500 : GOSUB 6900
 6530 M$=MID$(A6$,2,1) : N$=LEFT$(A6$,1)
+RIGHT$(A6$,4)
 6540 A5$=N$ : GOSUB 5500 : GOSUB 6900
 6560 M$=MID$(A6$,3,1) : N$=LEFT$(A6$,2)
+RIGHT$(A6$,3)
 6570 A5$=N$ : GOSUB 5500 : GOSUB 6900
 6590 M$=MID$(A6$,4,1) : N$=LEFT$(A6$,3)
+RIGHT$(A6$,2)
 6600 A5$=N$ : GOSUB 5500 : GOSUB 6900
 6620 M$=MID$(A6$,5,1) : N$=LEFT$(A6$,4)
+RIGHT$(A6$,1)
 6630 A5$=N$ : GOSUB 5500 : GOSUB 6900
 6650 M$=RIGHT$(A6$,1) : N$=LEFT$(A6$,5)
 6660 A5$=N$ : GOSUB 5500 : GOSUB 6900
 6680 RETURN
 6800 :
 6830 REM  SUB 6900 REJOINS THE HELD-
 6840 REM  BACK LETTER WITH THE OTHER
 6850 REM  5 FROM SUB 5500.
 6860 :
 6900 T=0   :   II=0    :    PRINT "•"
 6910 FOR K=1 TO 120
 6920 : T=T+1 : A$=M$+A5$(K) : PRINT A$,
 6940 : IF T=4 THEN T=0 : II=II+1 : PRIN
T
 6960 : IF II<>11 THEN 6980
 6970 : PRINT "TYPE  OK  TO CONTINUE ",
: INPUT#1,B$
 6375 : GOSUB 9000 : PRINT : T=0 : II=0
 6977 : PRINT "•"
 6980 NEXT K
 6985 PRINT "TYPE  OK  TO CONTINUE "; :
INPUT#1,B$
 6987 GOSUB 9000
 6990 RETURN
 7000 :
 7020 :
 8000 PRINT "↓↓TYPE  1  TO CONTINUE, 2
TO END"
 8020 INPUT W : IF W=1 THEN 240
 8040 END
 8100 :
 8120 :
 9000 IF B$="STOP" THEN 8040
 9020 RETURN
READY.
```

memory. Instead, only 120 are saved at a time. These 120 memory locations are written over after they have been printed on the screen.

So How's That Easy?

By going back and forth between all the subroutines, the computer executes approximately 3300 statements to process a six-letter word. If you wanted to write the program for maximum computer efficiency, you could have the program divide the word into its six individual letters and then go through the 720 statements that represented each possible combination. This means the computer could probably get away with only 750 executions, which is much better than 3300.

Keep in mind, however, that you are the one typing in these 750 statements to save your computer a few seconds. With the subroutine method you can get by with only about 150 statements and can process three-, four- and five-letter words as well. You paid a lot of money for the unit. You might as well let it do most of the work.

Many parts of this program are similar. Some programmers do this because it makes it easier to debug; others do it because it makes it easier to follow the program; some because if one part works, chances are they'll all work. I do it because I'm lazy.

For instance, lines 5570 and 5600 are exactly the same. Using the PET's editing features you only need to list line 5570, edit the line number to 5600 and hit the return key. Line 5570 isn't affected, and line 5600 has been written. Line 5580 needs only a few modifications to produce line 5610. The REM statements preceding subroutines 5500 and 5900 are only slightly different from those explaining subroutines 4500 and 4900, or even 3500 and 3900. Even the numbers aren't hard to change.

Naturally, this has to be planned before you start to key in your statements. Also, this type of editing requires concentration. I list the statements frequently to be sure I'm doing it right.

Is It Worth It?

If you've read this far because you're lazy too, you might be wondering if the program is worth the time it takes to type it in.

Well, it is fun, and it certainly makes short work of the scrambled word games in the newspaper's puz-

zle section. However, I doubt anyone would let you use a computer against them at Scrabble.

The biggest value in this program is for those who have never used nested subroutines. By adding your own statements, you can trace the action of the program. 3510 PRINT "SUB 3500" will let you know each time subroutine 3500 is called. 3510 S3 = S3 + 1 will keep count of how many times it is used. Your own imagination will let you come up with more examples.

Acknowledgements

One good idea in programming is to borrow good ideas from others. Scramble opens file 1 and uses INPUT#1 instead of INPUT for user response to the PET's request for input. This was used by Carol Ascolillo and Nancy Schwartz in their article "Cover Up" (*Microcomputing*, August 1979, p. 26). It keeps the program from ending prematurely if the RETURN key is hit with no input. The program listing uses the concept of prettyprinting as explained by Wallace Kendall's article "Prettyprinting with Microsoft BASIC" (*Microcomputing*, May 1979, p. 80).■

Variables:	
A3$(6)	one of six possible rearrangements of a three-letter word
A4$(24)	one of 24 possible rearrangements of a four-letter word
A5$(120)	one of 120 possible rearrangements of a five-letter word
A6$(120)	one of 720 possible rearrangements of a six-letter word
W$	word to be juggled
L	number of letters in the word to be juggled
A3$	a three-letter word to be juggled in subroutine 3500
A4$	a four-letter word to be juggled in subroutine 4500
A5$	a five-letter word to be juggled in subroutine 5500
A6$	a six-letter word to be juggled in subroutine 6500
A$	first letter of a three-letter word to be juggled in subroutine 3500
B$	second letter of a three-letter word to be juggled in subroutine 3500
C$	third letter of a three-letter word to be juggled in subroutine 3500
D$	one letter of a four-letter word kept back in subroutine 4500. The other three go to subroutine 3500
E$	three letters sent from subroutine 4500 to subroutine 3500
G$	one letter kept back in subroutine 5500 while the other four go to subroutine 4500
H$	four letters sent from subroutine 5500 to subroutine 4500
M$	one letter kept back in subroutine 6500 while the other five go to subroutine 5500
N$	five letters sent from subroutine 6500 to subroutine 5500
I1	lower limit in a FOR-NEXT loop
I2	upper limit in a FOR-NEXT loop
T	=1 to 4; the number of words printed across the screen
II	=1 to 11; the number of rows of words
J	subscript of letters returning from rearrangement in a lower subroutine
W	a number input to END or CONTINUE
Graphics:	
•	clear screen
↓	cursor down

Table 3. Symbols.

THE ORIGINAL MAGAZINE FOR OWNERS OF THE TRS-80™* MICROCOMPUTER

OFTWARE
OR TRS-80™
OWNERS

THE COMPUTRONICS INC.

MONTHLY
NEWSMAGAZINE
FOR TRS-80™
OWNERS

MONTHLY NEWSMAGAZINE

Practical Support For Model I, II & III

- PRACTICAL APPLICATIONS
- BUSINESS
- GAMBLING • GAMES
- EDUCATION
- PERSONAL FINANCE
- BEGINNER'S CORNER
- NEW PRODUCTS
- SOFTWARE EXCHANGE
- MARKET PLACE
- QUESTIONS AND ANSWERS
- PROGRAM PRINTOUTS

. . . . AND MORE

PROGRAMS AND ARTICLES PUBLISHED IN OUR FIRST 12 ISSUES INCLUDE THE FOLLOWING:

- A COMPLETE INCOME TAX PROGRAM (LONG AND SHORT FORM)
- INVENTORY CONTROL
- STOCK MARKET ANALYSIS
- WORD PROCESSING PROGRAM (FOR DISK OR CASSETTE)
- LOWER CASE MODIFICATION FOR YOUR VIDEO MONITOR OR PRINTER
- PAYROLL (FEDERAL TAX WITHHOLDING PROGRAM)
- EXTEND 16 DIGIT ACCURACY TO TRS-80™ FUNCTIONS (SUCH AS SQUARE ROOTS AND TRIGONOMETRIC FUNCTIONS)
- NEW DISK DRIVES FOR YOUR TRS-80™
- PRINTER OPTIONS AVAILABLE FOR YOUR TRS-80™
- A HORSE SELECTION SYSTEM***ARITHMETIC TEACHER
- COMPLETE MAILING LIST PROGRAMS (BOTH FOR DISK OR CASSETTE SEQUENTIAL AND RANDOM ACCESS)
- RANDOM SAMPLING***BAR GRAPH
- CHECKBOOK MAINTENANCE PROGRAM
- LEVEL II UPDATES***LEVEL II INDEX
- CREDIT CARD INFORMATION STORAGE FILE
- BEGINNER'S GUIDE TO MACHINE LANGUAGE AND ASSEMBLY LANGUAGE
- LINE RENUMBERING
- AND CASSETTE TIPS, PROGRAM HINTS, LATEST PRODUCTS COMING SOON (GENERAL LEDGER, ACCOUNTS PAYABLE AND RECEIVABLE, FORTRAN 80, FINANCIAL APPLICATIONS PACKAGE, PROGRAMS FOR HOMEOWNERS, MERGE TWO PROGRAMS, STATISTICAL AND MATHEMATICAL PROGRAMS (BOTH ELEMENTARY AND ADVANCED) AND

* TRS-80™ IS A TRADEMARK OF TANDY CORP

REE*

WORD PROCESSING PROGRAM For writing letters, text, mailing lists, etc., with each new subscriptions or renewal.
LEVEL II RAM TEST Checks random access memory to ensure that all memory locations are working properly.
DATA MANAGEMENT SYSTEM Complete file management for your TRS-80™.
CLEANUP Fast action Maze Game.
ADVENTURE Adventure #0 by Scott Adams (From Adventureland International).

FREE

*** All programs are supplied on cassette (add $3 for Diskette Version - add $5 for modified Mod-II Version).**

SEND FOR OUR NEW 48 PAGE SOFTWARE CATALOG (INCLUDING LISTINGS OF HUNDREDS OF TRS-80™ PROGRAMS AVAILABLE ON CASSETTE AND DISKETTE). $2.00 OR **FREE** WITH EACH SUBSCRIPTIONS OR SAMPLE ISSUE.

✓6

COMPUTRONICS INC.
ATHEMATICAL APPLICATIONS SERVICE™

50 N. PASCACK ROAD
SPRING VALLEY, NEW YORK 10977

master charge

24 HOUR ORDER LINE
(914) 425-1535

VISA

NEW TOLL-FREE ORDER LINE
(OUTSIDE OF N.Y. STATE)
(800) 431-2818

ONE YEAR SUBSCRIPTION $24 ....
TWO YEAR SUBSCRIPTION $48 ....
SAMPLE OF LATEST ISSUE $ 4 ....
START MY SUBSCRIPTION WITH ISSUE ....
(#1 - July 1978 • #7 - January 1979 • #12 - June 1979 • #18 - January 1980)
NEW SUBSCRIPTION .... RENEWAL ....

CREDIT CARD NUMBER ______________________ EXP. DATE ____________

SIGNATURE ______________________

NAME ______________________

ADDRESS ____________ CITY ____________ STATE ____________ ZIP ________

*** ADD $6 YEAR (CANADA, MEXICO) - ADD $12 YEAR AIR MAIL - OUTSIDE OF U.S.A., CANADA & MEXICO ***

This installment contains a miscellany of short, but no less significant, SWTP thoughts.

Thoughts on the SWTP Computer System

By Peter A. Stark

Several readers have asked for a tutorial on 6800 machine- and assembly-language programming. Recent installments of the Kilobaud Klassroom (November and December 1980), which features a construction series using the 6802 processor, cover machine- and assembly-language fundamentals. Since the 6802 has the same language as the 6800, you will find that almost all of the material there fits your SWTP system directly.

Serial Port Hand-Shaking

We normally think of serial interfaces as operating open loop—that is, sending out data to a printer or terminal regardless of whether this device is ready to receive that data or not. This is the normal state of affairs.

But there are many new printers that have a serial port capable of running at a higher baud rate than the printer can really print. In this case running open loop is almost guaranteed to lose characters. We therefore need some way of closing the loop—that is, providing a feedback signal from the printer back to the computer to slow down data when the printer cannot keep up.

Here is a typical case. There are several new printers in the \$600–\$900 range rated at about 120 characters per second. You would therefore think that 1200 baud would be an ideal speed, since this represents exactly 120 characters per second. Such is not the case.

In many cases, the print head makes a complete pass from the left margin to the right margin (or back, in the case of printers which print bidirectionally), regardless of how many characters are to be printed on that line. If the paper is 120 characters wide (to keep it simple), then that pass takes one second, regardless of how many characters are actually printed.

If the printed line has 120 characters, then they will all be printed in one second. But if the line has only ten characters, then this will still take one second, and hence the printing speed is only ten characters per second. Here, then, is a difference between dumb printers and smart printers—the smart printer will move its print head only part way across the paper if the printed line is short, and thereby achieve a faster transfer rate. Really good printers—generally in the \$1300–\$1500 range—will have a fast slew speed so that the print head will speed up when moving from one place to another, and may even have a fast up-down paper movement speed.

In any case, what baud rate should we use to feed a dumb printer? Even if we use a baud rate sending just ten characters per second, then lines longer than ten characters will be sent slow enough, but shorter lines will still give us problems.

Most such printers provide an internal FIFO (first in, first out) buffer of anywhere from 256 to 2048 bytes, which provides temporary storage in case the input data rate gets slightly ahead of the actual printing speed. Nevertheless, on long printouts of short lines, it is still possible to lose characters.

Thus the need for closing the loop. This turns out to be very simple when using the MP-S ACIA-type interface.

Most such serial printers provide a hand-shaking output called BUSY, READY, or something like that, which indicates when the printer cannot accept data. For example, the MPI 88T printer I worked with provides an output called BUSY, which is low (near 0 V) when the printer is ready for more data, but goes high (near +5 V) when its buffer fills up and cannot accept more.

The 6850 ACIA, on the other hand, has a Clear to Send ($\overline{CTS}$) input on pin 24, which permits the ACIA to output serial data when low, but halts output when high. All we have to do is connect the printer's BUSY signal to pin 24 of the ACIA.

The actual connection is just a bit tougher. On the MP-S board, ACIA pins 23 and 24 are both grounded, and it is difficult to break the foil leading to pin 24 to open up that pin. I solved that by bending pin 24 out of the socket, slipping a single Molex pin over that IC pin and soldering a thin jumper to the tip of the Molex pin. This makes the entire change reversible if needed. I used the index pin on the connector at the top of the MP-S board to bring the BUSY signal to the board.

Once this is done, you can run the printer and interface at 1200 or even 9600 baud, and never lose data. The computer will output rapidly to the printer until it fills up the buffer, and then slow down to precisely the right

Peter A. Stark (PO Box 209, Mt. Kisco, NY 10549) is a professor of electrical and computer technology at Queensborough Community College, City University of New York, Bayside, New York. He has contributed many pieces to Microcomputing *including his Kilobaud Klassroom articles and the SWTP series.*

speed to keep the buffer full, while the printer runs at its maximum speed.

One potential problem is that MOS ICs such as the ACIA are prone to static electricity damage. Connecting pin 24 of the ACIA right out like this may lead to potential problems if static electricity gets to this lead. A safer solution would have been to buffer this input with a TTL device (such as two inverters in a 74LS04, back to back). Being in a hurry, I chose not to do this, but it is something to keep in mind.

9600 Baud

We've mentioned this once before—the SWTP motherboard only carries up to 1200 baud clock to the interfaces, but baud rates up to 9600 are available at the MC14411 clock generator on the CPU board. To make the switch, simply cut the 150 baud output at the 14411 (which you will never use anyway) and substitute the baud rate you want. This will place the required baud rate on the 150-baud line properly buffered on the CPU board, and you need just connect to the 150-baud line on the interface.

Converting an MP-C to an MP-S

Many users have an MP-C PIA-type serial interface left over from their MIKBUG days, having replaced it with an MP-S. Except for the fact that an MP-C uses a PIA, whereas the MP-S interface uses an ACIA, the two interfaces are very similar to each other. With just a little work (and about $5 for a new 6850 ACIA and some thin wire), you can update an older MP-C interface into an MP-S as follows:

1. If the ICs on your MP-C board are in sockets, then unplug IC1, IC3, IC4 and IC6, discard and go to step 2. If the ICs are not in sockets, then a little preparatory work is needed. IC3, IC4 and IC6 will not be used; thus, you can cut their pins with a pair of diagonal cutters, and simply discard each IC, leaving the pins soldered in place. IC1, the PIA, will also be discarded, but an ACIA socket has to be installed in its place; thus, it has to be removed neatly. The easiest method is to cut all its pins, take the IC off and then carefully—using a vacuum-type solder-sucker—remove the solder from each pin and pull out the pin. After the PIA has been removed, install a 24-pin socket for the ACIA in the PIA position, being sure that pin 1 of the socket is in the hole formerly occupied by PIA pin 6.
2. Remove the C jumper.
3. Next, we will modify some of the wiring around the PIA (IC1) position to accommodate the ACIA. Use a single-edge razor blade or X-acto knife to cut the following lands. The best method is to make two cuts about 1/16 inch apart (close to the IC1 socket), and then use the tip of a small soldering iron to pry up and remove the piece of copper between the lands. Then clean the cut area with a Q-Tip and some alcohol.
4. On top of the board, cut the land leading to PIA pin 34.
5. On the bottom of the board, cut the lands leading to PIA pins 9, 24 and 35, close to the socket.
6. Now use some thin wire (the type used for wire-wrapping is best) to make the following connections on the bottom of the printed circuit board:

PIA pin 6 to PIA pin 1 (ground)
PIA pin 7 to IC5 pin 8 (received data)
PIA pin 8 to IC3 pin 1 (clock)
PIA pin 9 to PIA pin 8 (clock)
PIA pin 10 to PIA pin 19 (RTS)
PIA pin 11 to IC5 pin 13 (transmitted data)
PIA pin 12 to PIA pin 38 ($\overline{IRQ}$)
PIA pin 13 to PIA pin 15 (CS0 to +5 volts)
PIA pin 14 to PIA pin 23 (CS2 to I/O Select)
PIA pin 15 to PIA pin 17 (CS1 to +5 volts)
PIA pin 16 to PIA pin 36 (RS0)
PIA pin 17 to PIA pin 20 (Vcc)
PIA pin 24 to PIA pin 21 (R/$\overline{W}$)
PIA pin 34 to ground
PIA pin 35 to ground

7. Now plug the ACIA into its socket, taking care to position it so its pin 1 is in the hole formerly occupied by pin 6 of the PIA.

This completes conversion of the MP-C board to the MP-S circuit. All connections between the card and the terminal remain the same as before.

Though the ACIA can work at higher baud rates, the MP-C printed-circuit board only has positions for 110- or 300-baud jumpers. If you want to operate at higher rates, you will have to connect the baud rate jumper directly to the Molex connector on the bottom of the MP-C board. Look at the board layout for the motherboard to see which pin has which baud rate signal on it.

The 6875 Clock Generator

One of our readers mentioned that Motorola has stopped producing the 6875 clock generator used on the MP-A2 CPU board. If you anticipate needing one, you should make the rounds of Motorola distributors now, while some may still be on the shelves.

This IC is also used in the Heath ET-3400 trainer; it may be available from Heath as a last resort if you need it (part number 443-840). After that, if your 6875 goes, it may be necessary to replace the entire clock circuit with the one used on the MP-A board, or replace the 6800 with a 6802 and change a few connections (most of the 6802 pins are positioned the same as those of the 6800).

CT-82 Modifications

SWTP has released a new control ROM ($30) which fixes up some previous bugs and also adds new functions. This should be a very worthwhile addition to an excellent terminal.

SWTP BASIC 3.0 under Flex 2.0

Until now, the only BASIC available for Flex 2.0 has been TSC BASIC and TSC Extended BASIC. This makes the conversion from MiniFlex

to Flex 2.0 quite an expensive proposition, especially since the standard TSC BASIC has too little accuracy for financial calculations, while the Extended BASIC requires almost 20K just for the interpreter.

An article in the July 1980 issue of *68 Micro Journal* by Richard G. Cagle (Applevalley Day School, 11103 Sagepark Lane, Houston, TX 77089) suggests an alternative.

Richard's article shows how to modify SWTP Disk BASIC version 3.0 (which was originally supplied with SWTP disk systems as part of the miniFlex package) to run under Flex 2.0. He developed the modification to permit some business software he is marketing to run under Flex 2.0 with minimal modifications.

Though SWTP BASICs are much, much slower than TSC BASICs, this modification should be of interest to those who would like to switch to Flex 2.0 but don't want to invest heavily in a new BASIC.

BASIC Differences

TSC BASICs handle null strings slightly differently from older BASICs.

First, TSC BASIC assumes that the null string is the same as a single space. For example, if you try running

```
IF ""=" " THEN PRINT "SAME"
```

you will get the printout SAME in TSC BASIC, but not in SWTP BASICs.

Second, in SWTP BASIC you can enter a null string during an INPUT statement. For example, in response to

```
INPUT A$
```

you can just hit a carriage return, and SWTP BASIC will make A$ equal to a null. TSC BASIC does not permit that—you must actually enter one or more characters. This is unnerving if your program contains something like

```
INPUT "HIT RETURN TO CONTINUE",A$
```

TSC BASIC will refuse to accept just a return.

TSC BASICs also insert extra spaces before numbers; this sometimes causes problems.

An important fact to remember when using sequential disk files is that TSC BASICs do not automatically insert delimiters between fields. For example, suppose that we have the strings A$="ABC", and B$="DEF". If we write these to a disk file in SWTP BASIC, the file will have the characters

```
ABC,DEF carriage return.
```

BASIC inserts a comma between the two strings, and a carriage return at the end. But TSC BASIC omits the comma. Thus, if we read this file back into A$ and B$, SWTP BASICs will have A$ equal to ABC, and B$ again equal to DEF; TSC BASIC, on the other hand, will have ABCDEF in A$ and nothing in B$.

This is important to remember if you are transferring a program from an earlier BASIC to TSC BASIC.

The Abort Switch

Have you ever looked at a Motorola Exorciser system? This is a development system for products using Motorola processors. One interesting aspect of this computer is that on the front panel, right next to the reset switch, is another switch labelled abort. What does it do?

The abort button generates an NMI interrupt. When this occurs, the Exorciser aborts whatever program it is currently executing, stores register contents on the stack, prints the register dump on the terminal and returns to the monitor.

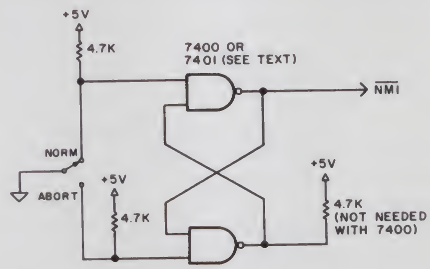


Fig. 1. Abort switch debouncing circuit.

In many ways this is like a reset. But the register dump also tells us where the computer was and what it was doing when it was interrupted. After a program is aborted, it can then be modified and restarted, just as if it had just hit a breakpoint. This is an extremely worthwhile feature—how many times has your computer gone into never-never-land leaving you wondering where it was? With the abort button you would have known.

Adding an abort switch to the SWTP system is not that hard; it involves some hardware and some software.

Connecting a push-button switch from the NMI line on the bus to ground would be good enough to generate an NMI, but switch bounce would cause several successive NMIs to be generated; this would overflow the stack and cause problems. Thus, we need to debounce the switch.

Fig. 1 shows a debouncing circuit which requires an SPDT push button. A 7400 NAND gate package (without the output resistor) can be used if you have no other sources of NMI interrupts; otherwise, a 7401 open-collector NAND gate (with resistor) is required to leave the NMI line available for other uses.

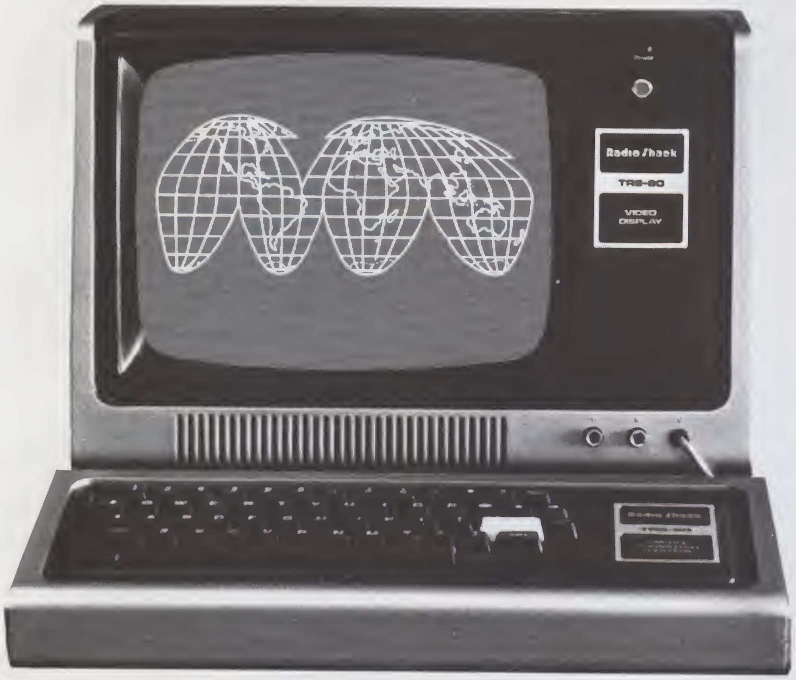
The quickest software fix to implement the abort switch is to use the monitor's M command to read out the address from the SWI vector (location A012) and place the same address into the NMI vector (location A006). In the case of SWTBUG, place the address E123 into A006; with other monitors the address may be different. (Of course, you must remember to do this before executing a program you may want to abort later. If you haven't prepared for it, then you cannot abort later.) With this change, pushing the abort will act the same as encountering a breakpoint in the program, resulting in a return to monitor and a register printout.

Though this modification will work, it has two drawbacks. First, the program counter (PC) address dumped will be incorrect, and second, under some conditions it may not be possible to restart the program.

On a breakpoint, the PC address in the stack points to the next instruction after the SWI (3F) which caused the breakpoint. SWTBUG subtracts 1 from this, so that the PC address in the register dump points to the breakpoint location, rather than past it.

On an NMI or abort, the PC address in the stack points to the next instruction after the one which was executing when the interrupt occurred. After SWTBUG subtracts 1 from this, the displayed PC address will point to the last byte of the last instruction. This may be the instruction code itself (in one-byte instructions), or the second or third bytes of this instruction. Thus, some interpretation of the register dump is required.

The other problem is this—when SWTBUG finishes the register dump, it checks whether a breakpoint was in effect. If it was, then it keeps the current stack pointer; otherwise, it resets the stack pointer to A042. Thus, it may change the stack pointer after an abort NMI, and then you cannot continue the aborted program. This is normally not much of a prob-



CompuServe's information service.

A world of information available. Right now.

If you have a personal computer—or a computer terminal — CompuServe can bring a world of information into your home or small business.

CompuServe
CompuServe is a major computer services company that has been serving top industrial and governmental clients for more than a decade. Now we're also applying our extensive computing capacity to the delivery of information services to your personal computer.

The Information Service
This exciting service is available in more than 250 cities between 6 p.m. and 5 a.m. weekdays, all day weekends and most holidays. Cost? Only $5.00 per hour billed in minutes. All you need is a 300 baud modem, and our complete information service is only a local phone call away.

Welcome to CompuServe's information service.

- *News. Weather. Sports.* Major regional newspapers. Plus international news services.
- *Finance. MicroQuote.* Updates and historical information on stocks, bonds and commodities.
- *Entertainment.* Theatre, book, movie and restaurant reviews. Plus opera, symphony, ballet, dance, museums, galleries...
- *Electronic Mail.* Create, edit, send and receive messages from any other CompuServe user ... nationwide.
- *Home & Educational Reference Service.* Anything you want to know ... from encyclopedia information to household tips.
- *CompuServe User Information.* In case you need technical help ... and information on new services as they become available.
- *MicroNET.* For the computer hobbyist. Software Exchange, line printer art gallery, challenging games, programming languages, word processing, business & educational programs ... and much, much more.

And this is just the beginning. CompuServe is continually adding to its list of available services. The world of electronic information isn't coming tomorrow. It's here today.

CompuServe is Available at Radio Shack® Stores
Your local Radio Shack® Store Sells CompuServe's information service. There are more than 6,000 Radio Shack® Stores and Dealers nationwide. Check with the outlet nearest you.

Radio Shack is a division of Tandy Corporation.

CompuServe

Information Service Division ✓147
5000 Arlington Centre Blvd.
Columbus, Ohio 43220
(614) 457-8600

✓ Reader Service—see page 226

lem, since we are usually more interested in knowing what was going on, rather than restarting.

As an aside, the HUMBUG monitor we described in past installments has the correct NMI instructions to provide the correct PC address during an abort.

Unfortunately, the abort switch will sometimes not work. If a program bombs sufficiently that it erases all of memory, then it may also erase the NMI vector at A006; in that case a reset is the only alternative. Placing the NMI vector directly into the monitor ROM would be an alternative, but this would prevent the use of NMI for other purposes.

Finally, both Flex 2.0 and SSB DOS use the NMI vector for their own purposes. Hence the abort switch cannot be used while they are running. If you exit the DOS to execute a program, you should remember to restore the NMI vector prior to using it, since it was changed by the DOS.

Power Supply Modifications

After adding nine 2708 EPROMs to my system, the voltage on my +12 and −12 V power supplies started to sag below the desirable level, and the main power transformer seemed to run hotter than it used to. Time for another update.

My solution was simple. First, I disconnected the two red wires from the transformer to the SWTP power supply board and taped them. This removed the ±12 V supplies from the main power transformer and decreased the power transformer load by some 25 watts, resulting in cooler operation.

Then I connected a Radio Shack 25.2 V center-tapped transformer (catalog number 273-1512) to the power supply board, with the two red wires going to the two terminals which earlier had the two red wires removed, and the yellow wire to ground (120 V going to the black wires, of course).

This increased the supply voltage from +12 and −12 V to about 14–15 V, which provides a better safety margin.

The only thing to watch out for is that some boards require +12 or −12 V, and do not use a regulator because they count on the supply voltage being very close to that value. These boards must be modified by installing a 7812 regulator for +12 V, or a 7912 regulator for −12 V. (Note, however, that the MP-C and MP-S boards do not require an exact voltage, and the regulator is not needed even with 14–16 V.)

6809-Based Bulletin Board

What may be the first 6809-based community bulletin board system (CBBS) was started this past summer by a 6800/6809 computer dealer in Pennsylvania.

As Tom Quay of Lehigh Computer Works (1132-2 Tilghman, Allentown, PA 18102) put it, the purpose of the bulletin board was to serve as a communications medium between his customers. 6800/6809 users can share ideas, find out about the latest software and hardware, place inquiries or orders with the store or receive answers to their questions.

The interesting thing about the system is that it uses no special software. The system uses an auto-answer modem which takes care of answering and hanging up the phone automatically, without involving the

computer at all.

The system is essentially a standard SWTP system with two disks, running TSC's Flex. Only three disk utilities are on the disk—CAT, BUILD and LIST. Moreover, a minor modification to Flex has disabled the GET and MON commands. Thus, anyone familiar with Flex can use the system without any further instructions.

Since only the three functions are available, there is little chance of a user abusing the system. It is possible to get a catalog of disk files on either drive, to build a new text file or list an existing file, but not to return to the monitor, delete files or take over control of the system. Tom reads all files every day or so, and simply removes any files no longer needed. Altogether, it is a simple system that could be implemented by almost any 6800 or 6809 owner.

In the first few days of operation, Tom did discover a few problems which had to be solved. For example, if a user asked for a catalog of drive 3 (on a two-drive system), Flex would hang up; obviously the system had to be patched to prevent accessing nonexistent drives. But on the whole, the entire idea is quite novel.

If you want to access the system, the phone number is 215-437-0556, which is used by the store staff during the day. The CBBS is operational from 9 PM to 8 AM.

Video Status

There are several video boards available for SS-50 systems:

- Gimix, Inc. (1337 West 37th Place, Chicago, IL 60609), has a standard video board and a high-resolution graphics card set.
- Thomas Instrumentation (168 Eighth Street, Avalon, NJ 08202) has an inexpensive video board, available assembled or as a bare board.
- Percom Data Co. (211 N. Kirby, Garland, TX 75042) has a video board called the Electric Window, as well as a color video board called the Electric Crayon (which is intended for the TRS-80).
- Johnson Micro Computer (2607 E. Charleston, Las Vegas, NV 89104) does not make a video board as such, but they make an adapter board for a CT-64 terminal which connects it to an SS-50 SWTP computer so that the terminal's screen memory becomes directly addressable by the computer.
- F&D Associates (1210 Todd Road, New Plymouth, OH 45654) has three video boards—two for black and white text and/or graphics and one for color. They sell bare boards and instructions for building your own.

There are various differences between the boards in terms of both the hardware they use and the software needed to drive them.

Some boards (for example, the Johnson and Thomas boards) use a minimum of hardware. The boards have 1K of RAM, which represents 1K of characters to be displayed, generally in 16 lines of 64 characters. Each memory location corresponds to a particular place on the screen; put an ASCII character there (with an STA in machine language or a POKE in BASIC) and the character appears in the corresponding place on the screen. This approach is very simple, but it requires additional software to keep track of characters. For example, when the bottom line on the screen is full, it is necessary to erase the top line, move all remaining lines up one position and then erase the bottom line in preparation for new text. This is called scrolling, and must be done by software on these simple boards.

Other boards (such as the Gimix and Percom) use an intelligent CRT controller which does most of the scrolling internally. On these boards, the text need not physically be moved inside memory to move one line up on the screen; the controller simply readdresses each line when it reads out the characters being displayed, so that lines appear to move up without actually changing place in memory itself.

The latter approach simplifies the software required to some extent, but it also has a further advantage. When scrolling is done by software, physically moving all characters up by one line in memory takes time. Not only does this slow down the operation of the system, but during this movement the memory is unavailable to the display circuitry (since the computer is using it) and thus we get interference on the screen.

From the above discussion, you might surmise that the hardware approach, using an intelligent controller, is better than the software approach. Actually, there are other factors to be considered. If you want to use the screen to display graphic symbols, you need to know exactly which RAM location corresponds with which spot on the screen; this is difficult to do with those boards where a controller does the scrolling. Moreover, if you want to split the screen into portions, or reserve a line or two for a special status display, then again the controller tries to defeat your efforts.

For this reason, it is sometimes better to defeat the controller on the board and go back to using the software scrolling approach. To illustrate, let's discuss the Percom Electric Window video board and the program of Listing 1.

The Percom board uses the CRT 5027 video timer-controller made by Standard Microsystems Corp. (35 Marcus Blvd., Hauppage, NY 11787). This is a fairly complex circuit which permits a wide variety of formats having from one to 64 lines and one to 200 characters per line. The most convenient display format on the Percom board, however, is 16 lines of 80 characters; a 24-line format is possible, but requires a long-persistence screen to avoid an unpleasant flicker.

In the Percom board, the screen RAM is at locations D800 to DFBF, and the controller registers are at DFC0 through DFCF; addresses DFD0 through DFFF are unused.

When the controller is set up for a 16×80 display, the screen RAM is divided up so that the first 80 locations (D800–D84F) are called line 0, the next 80 locations (D850–D89F) are line 1, and so on, up through locations DCB0–DCFF, which are line 15.

These line numbers, however, do not correspond with the actual lines as seen on the screen; instead, we can specify which of these lines should become the bottom line on the screen by placing that line's number into location DFC6 (called BOTLNE in the program). Thus, if we put 15 (or hex 0F) into BOTLNE, then line 0 will be at the top while line 15 is at the bottom. But if we put 00 into BOTLNE, then line 0 will be at the bottom, and the next line, line 1, will start off at the top. This controller scrolls the screen simply by incrementing BOTLNE by 1, going from 00 through 0F and then back to 00. This rearranges the text lines on the screen without actually moving contents of memory.

To keep track of the cursor—and the location where the next character should go—we have locations DFCD (called CURV) and DFCC (called CURH). These locations keep track of the vertical and horizontal position of the cursor. (There is a slight inconsistency which should be explained—CURH assumes that there is a two-character margin to the left of the screen, so CURH=2 is actually the left-most character on the screen. This explains why the program sets CURH to 2, or subtracts 2 in several places.) Percom video driver software, called WINDEX, normally uses CURV and CURH to position the next character and uses BOTLNE to have the 5027 controller scroll the screen without actually moving screen contents.

As mentioned earlier, the disadvantage of this system is that drawing pictures, or reserving lines for special purposes, is difficult in this kind of a scheme. I devised the program in Listing 1, which is loosely (very loosely) based on Percom's Windex driver. This video driver program uses strictly software to place characters on the screen.

In this scheme, the screen RAM at D800–DCFF strictly corresponds with screen position. D800 is the character at the top left of the screen, D801 is just to its right and so on, all the way to DCFF, which is at the lower right corner. This is achieved by forcing BOTLNE to always equal 0F, so that line 15 is always at the bottom of the screen and never allowed to move. It is therefore possible to poke characters or graphic symbols into the screen RAM, and they will always appear exactly where you place them (unless scrolled later).

Another useful aspect of the program is the use of the variable TOP to define the top of the usable screen memory. TOP is set to D800 during the initialization part of the program, so that normally the entire area from D800 through DCFF is used for the 16-line display. But TOP can be initialized above D800, thereby reserving an area from D800 to TOP so that it is not used by the video driver.

For example, suppose we set TOP equal to D850. The driver will now ignore the first 80 locations of the screen RAM, corresponding to the top line, line 0, on the screen. It will not place characters into this area, nor will it scroll up into it. Now you can put special text or graphics into this area and have it remain even while the rest of the screen is used. Anywhere from 0 to 255 locations can thus be reserved; reserving 80, 160 or 240 will reserve entire lines, while other numbers may reserve just parts of lines.

Such a reserved line on the screen is often called a status display, since it is used to show the status of the system. What kind of status?

Well, to begin with, it can display the time. If you have a complete clock, such as the JPC clock board, you can use interrupts to transfer the

Listing 1. Video board driver program.

```
                NAM   EW          DRIVER

          * VIDEO DRIVER FOR PERCOM VIDEO BOARD
          * LOOSELY BASED ON PERCOM'S "WINDEX" DRIVER
          * MODIFIED BY PETER A. STARK TO PERMIT
          * EASY DMA TO SCREEN MEMORY AND TO RESERVE
          * 0 TO 3 STATUS LINES AT TOP OF SCREEN

          * VIDEO BOARD REGISTERS
   (DCFF)   BOT     EQU   $DCFF       END OF SCREEN MEMORY
```

More

Reader Service—see page 226

Listing 1 continued.

```
  (DFCD)     CURV   EQU   $DFCD       CURSOR LINE REGISTER
  (DFCC)     CURH   EQU   $DFCC       CURSOR HORIZONTAL REGISTER
  (DFC6)     BOTLNE EQU   $DFC6       BOTTOM LINE

                    * OUTCHV - OUTPUT ON VIDEO BOARD
  (9000)            ORG   $9000
9000 37      OUTCHV PSH B             SAVE REGISTERS
9001 36             PSH A
9002 FF 9121        STX   SAVEX
                    * SEARCH CHARACTER TABLE FOR SPECIAL CASES
9005 CE 90D3        LDX   #CHTABL     POINT AT TABLE
9008 A1 00   TAB    CMP A 0,X         LOOK AT NEXT ENTRY
900A 26 04          BNE   TAB1        IF NOT FOUND
900C EE 01          LDX   1,X         FOUND SPECIAL CHAR
900E 6E 00          JMP   0,X         JUMP TO CORRECT ROUTINE
9010 08      TAB1   INX               POINTR AT NEXT ENTRY
9011 08             INX
9012 08             INX
9013 6D 00          TST   0,X         END OF TABLE?
9015 26 F1          BNE   TAB         NO, SO LOOK FOR MORE

                    * NOT SPECIAL CHARACTER, SO CONTINUE
9017 85 E0          BIT A #$E0        IS IT CONTROL CHAR?
9019 27 50          BEQ   EXIT        IGNORE IF YES

                    * NOW PROCESS PRINTABLE CHARACTER
901B 36             PSH A             SAVE CHARACTER
901C F6 DFCD        LDA B CURV        GET VERT POSITION OF CURSOR
901F CE D800        LDX   #$D800      POINT TO SCREEN MEMORY
9022 FF 911F        STX   POINTR
                    * NOW STEP FORWARD 80 BYTES FOR EACH LINE
9025 5A      STEP1  DEC B
9026 2B 0F          BMI   STEP3       QUIT WHEN B<0
9028 B6 9120        LDA A POINTR+1
902B 8B 50          ADD A #80         ADD 80 TO POINTER
902D 24 03          BCC   STEP2
902F 7C 911F        INC   POINTR      INCR ON OVERFLOW
9032 B7 9120 STEP2  STA A POINTR+1
9035 20 EE          BRA   STEP1       AND REPEAT
                    * NOW ADD 1 TO POINTER FOR EACH HORIZ CHAR
9037 B6 DFCC STEP3  LDA A CURH        ADD CURSOR HORIZ POSITION
903A BB 9120        ADD A POINTR+1    TO POINTER
903D 24 03          BCC   STEP4
903F 7C 911F        INC   POINTR      INCR ON OVERFLOW
9042 80 02   STEP4  SUB A #2          COMPENSATE FOR LEFT MARGIN...
9044 24 03          BCC   STEP5       ...(PECULIAR TO SMC CONTROLLER)
9046 7A 911F        DEC   POINTR      DECR ON BORROW
9049 B7 9120 STEP5  STA A POINTR+1
                    * NOW PLACE CHARACTER ON SCREEN
904C FE 911F        LDX   POINTR
904F 32             PUL A             RESTORE CHARACTER
9050 A7 00          STA A 0,X         PUT IN CORRECT SPOT
                    * CHAR IN CORRECT SPOT, UPDATE CURSOR
9052 7C DFCC FF     INC   CURH        MOVE CURSOR RIGHT
9055 B6 DFCC        LDA A CURH        CHECK POSITION
9058 81 51          CMP A #81         PAST END OF LINE?
905A 2F 0F          BLE   EXIT        NO, SO EXIT
                    * PAST END OF LINE, SO GO TO NEW LINE
905C 86 02          LDA A #2          MOVE CURSOR TO LEFT MARGIN
905E B7 DFCC        STA A CURH
9061 B6 DFCD LF     LDA A CURV        GET CURSOR VERT POSITION
9064 81 0F          CMP A #$F         BOTTOM OF LINE?
9066 27 09          BEQ   SCROLL      YES, SO GO SCROLL
                    * NOT YET AT BOTTOM
9068 7C DFCD        INC   CURV        MOVE CURSOR DOWN 1 LINE
906B FE 9121 EXIT   LDX   SAVEX       RESTORE REGISTERS
906E 32             PUL A
906F 33             PUL B
9070 39             RTS               AND EXIT

                    * ONCE AT BOTTOM LINE, SCROLL UP 1 LINE
9071 FE 9123 SCROLL LDX   TOP         POINT TO TOP OF USABLE RAM
9074 A6 50   SCROL1 LDA A 80,X
9076 A7 00          STA A 0,X         MOVE 80 LOCS FORWARD
9078 08             INX
9079 8C DCB0        CPX   #BOT-79     DONE SCROLLING?
907C 26 F6          BNE   SCROL1      NO
                    * ERASE BOTTOM LINE
907E 86 20   SCROL2 LDA A #$20
9080 A7 00          STA A 0,X
9082 08             INX
9083 8C DD00        CPX   #BOT+1      DONE?
9086 26 F6          BNE   SCROL2      NO
9088 20 E1          BRA   EXIT        AND EXIT

                    * ROUTINES FOR SPECIAL CHARACTERS

908A B6 DFCC BS     LDA A CURH        GET HORIZ POSITION
908D 81 02          CMP A #2          AT LEFT MARGIN?
908F 27 DA          BEQ   EXIT        YES, IGNORE
```

More

time at fixed intervals—1, 60 or 100 times per second—into the status portion of the video RAM. Even if you only have a timer board, such as the SWTP MP-T timer board, the interrupt program can still compute the time by counting up the number of interrupts and display it.

But there are other, more useful, things to display. Most of these essentially need a timer board such as the MP-T to periodically cause interrupts; each time an interrupt occurs, the interrupt service program gets some number from memory, converts it to ASCII and transfers it to some appropriate part of the status line. With care, several different status messages can be displayed in different portions of the line. Some of these might be:

- The contents of the BASIC or editor input buffer, showing a line exactly as it is currently being typed or edited.
- The track and sector currently being accessed by the disk.
- The number of characters read from or written to the disk since some previous time.
- The address of the most recently executed instruction in a machine-language program, or the line number of the most recently executed line in a BASIC program.
- The level—high or low—of one or more I/O lines, or the status of I/O equipment.
- The number of the port or ports currently active.

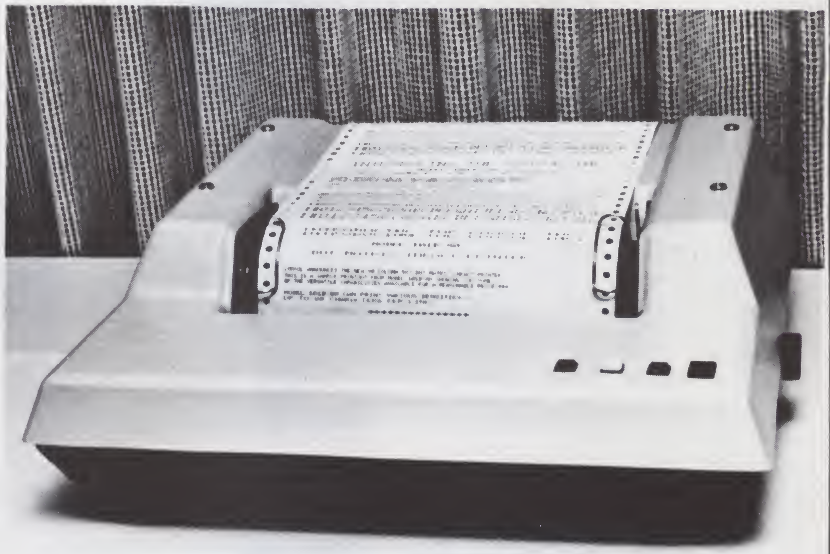
The possibilities are endless. For example, I was very intrigued recently by the status display of a friend's Alpha Micro computer. The first 77 spaces on that line were devoted to the 77 tracks on his disk drive; at each interrupt, the interrupt software would see which track had most recently been used, and place a period in the corresponding position on the status line. By watching the period move left and right, you could almost see the disk head moving in and out to various tracks.

A similar idea would be to use the first 16 positions for the 16 4K segments of memory, and put a period (or, better yet, a hex digit 0 through F) into the corresponding location to indicate the address of the most recent instruction. You could then observe programs as they wander through memory. I will be implementing this idea and will show how it is done in a future installment.■

Listing 1 continued.

```
9091 7A DFCC          DEC   CURH       MOVE CURSOR LEFT 1
9094 20 D5            BRA   EXIT

9096 B6 DFCD  VT      LDA A CURV       GET VERT POSITION
9099 81 00            CMP A #0         TOP OF SCREEN?
909B 27 CE            BEQ   EXIT       YES, IGNORE
909D 7A DFCD          DEC   CURV       MOVE UP 1 LINE
90A0 20 C9            BRA   EXIT       AND RETURN

90A2 86 02    CR      LDA A #2         SET CURSOR TO LEFT MARGIN
90A4 B7 DFCC          STA A CURH
90A7 20 C2            BRA   EXIT


90A9 8D 0F    CLEAR   BSR   HOMEUP     CLEAR SCREEN
90AB 86 20    ERASE   LDA A #$20       ERASE TO END OF FRAME
90AD FE 9123          LDX   TOP        START AT TOP OF USABLE SCREEN
90B0 A7 00    ERASE1  STA A 0,X        CLEAR NEXT
90B2 08               INX
90B3 8C DD00          CPX   #BOT+1     BOTTOM?
90B6 26 F8            BNE   ERASE1
90B8 20 B1            BRA   EXIT

90BA 86 02    HOMEUP  LDA A #2
90BC B7 DFCC          STA A CURH       SET CURSOR TO LEFT MARGIN
90BF 4F               CLR A            ALLOW FOR 1 OR 2 EMPTY LINES
90C0 F6 9124          LDA B TOP+1
90C3 C0 50    HOME1   SUB B #$50       COMPUTE TOP LINE
90C5 2B 03            BMI   HOME2
90C7 4C               INC A            MOVE DOWN TO NEXT
90C8 20 F9            BRA   HOME1
90CA B7 DFCD  HOME2   STA A CURV       SET CURSOR TO 1ST USABLE LINE
90CD 86 0F            LDA A #$F
90CF B7 DFC6          STA A BOTLNE     DISABLE AUTO SCROLLING
90D2 39               RTS


90D3 08       CHTABL  FCB   $8 BACKSPACE
90D4 90 8A            FDB   BS
90D6 0D               FCB   $D CARRIAGE RETURN
90D7 90 A2            FDB   CR
90D9 0A               FCB   $A LINE FEED
90DA 90 61            FDB   LF
90DC 10               FCB   $10 CLEAR SCREEN?
90DD 90 A9            FDB   CLEAR
90DF 0B               FCB   $B VERTICAL TAB
90E0 90 96            FDB   VT
90E2 0C               FCB   $C NON-DESTRUCTIVE SPACE
90E3 90 52            FDB   FF
90E5 7F               FCB   $7F RUBOUT
90E6 90 6B            FDB   EXIT
90E8 00               FCB   0

              * VINIT - VIDEO INITIALIZATION
90E9 4F       VINIT   CLR A
90EA CE DFC0          LDX   #$DFC0
90ED A7 0A            STA A $A,X       RESET CONTROLLER
90EF A7 0E            STA A $E,X       SET UP TIMING CHAIN
90F1 86 65            LDA A #$65
90F3 A7 00            STA A 0,X        HORIZONTAL LINE COUNT
90F5 86 64            LDA A #$64
90F7 A7 01            STA A 1,X        INTERLACE H SYNC
90F9 86 6D            LDA A #$6D
90FB A7 02            STA A 2,X        ROW SCAN CHAR
90FD 86 8F            LDA A #$8F
90FF A7 03            STA A 3,X        ROW FRAME
9101 86 03            LDA A #$03
9103 A7 04            STA A 4,X        LINE FRAME
9105 86 20            LDA A #$20
9107 A7 05            STA A 5,X        VERT START
9109 86 0F            LDA A #$0F
910B A7 06            STA A 6,X        LAST ROW
910D 4F               CLR A
910E A7 0A            STA A $A,X       RESET CONTROLLER
9110 A7 0E            STA A $E,X       START TIMING CHAIN
9112 86 D8            LDA A #$D8
9114 B7 9123          STA A TOP        SET TOP = D800
9117 7F 9124          CLR   TOP+1
911A 86 10            LDA A #$10       CLEAR SCREEN CODE
911C 7E 9000          JMP   OUTCHV     GO DO IT

              * VIDEO DRIVER STORAGE

911F          POINTR RMB    2          POINTER TO CURSOR LOC
9121          SAVEX  RMB    2          TEMP INDEX STORAGE
9123          TOP    RMB    2          TOP OF SCREEN MEMORY

                     END
```

COOSOL DATA LOGGER IMPACT PRINTER



FEATURES INCLUDE:

- Uses Standard Typewriter Ribbon
- Micro Processor Controlled
- Can Operate Continuously— No Thermal Problems (Has an all metal print head)
- 5 x 7 to Larger 10 x 7 and Larger 10 x 14 Dot Matrix Character Generator
- Standard 96 ASCII Character Font
- Upper and Lower Case Printing
- Up to 88 Characters Per Line
- Single Line Print Rate Is 160 CPS
- Average Print Rate Is 60 CPS For Ten Lines
- Graphics Capability With Extended Character Modes
- Programmable With 32 System Level Software Commands
- Standard Parallel and Serial Interface
- Software and Hardware Reset Interface
- Software Line Counting For Paging
- Baudrate Select From 110 to 9600
- Manual Paper Advance
- Manual Selftest and Reset
- Adjustable Tractor Width From 1 to 9½ Inches
- Interfaces with Apple, Atari, OSI, T.I., TRS-80 and the List Goes On

$495 Kit, 101B-80KE **$545 Assembled & Tested 101B-80E**

✓292

COOSOL, INC. P.O. BOX 743, ANAHEIM, CA 92805 (714) 545-2216 7 Days a Week

FOR THE APPLE :

µSPEED

- *Runs **six** to **sixty** times **faster** than Basic*
- *Programming capabilities well beyond Applesoft*
- *Auxiliary Processor (Am9511) for **fast** Floating Point*
- *High-speed, extended high resolution graphics*
- *Software development time cut in half*
- *Exceptionally compact, compiled code*
- ***Extensible,** structured language to meet **your** needs*
- ***Bonus:** Increased speed Applesoft functions*
- *Requires 48K Apple II or II+, single Disk*

The microSPEED Language System combines the benefits of compiled code and hardware arithmetic processing for the maximum in microcomputing performance. Developed from an extended version of Forth, the System is ideally suited to the serious Apple user.

THE POWERFUL, PAINLESS ALTERNATIVE TO BASIC AND PASCAL.

applied analytics incorporated

5406 Roblee Dr., Suite 701 Upper Marlboro, MD 20870 ✓134

Please send:
☐ Complete System $495.00
☐ User's Manual only $35.00
☐ Detailed information

Name ______________________

Address ______________________

City ____________ State ____________ Zip ____________

More power to you.

Disk drives, hardware and software— now more affordable *and* more available!

The choice is yours
Whether you need professional-looking cases and power supplies, complete disk drive packages or powerful, user-tested software, call A.M. Electronics. We manufacture and sell a complete line of affordable, high-quality and *readily-available* disk drive components and software to expand the capabilities of your TRS-80™ system.

Disk drive enclosures
All enclosures feature:

- One year power supply guarantee. Each unit is 100% tested, regulated and fused.
- Optimum venting for lower operating temperatures
- Excellent RF interference shielding. Steel covers come with lustrous dark grey finish.
- Built-in provisions for optional extender cable (5¼ only).

COMPLETE DISK DRIVE PACKAGES FROM $325!

5¼-inch drives
40-track MPI 51 w/case, power supply and extender cable **$325**
80-track MP 91 w/case, power supply and extender cable **$560**
Special Offer! 8-inch drives for Model I or II
Single Siemens drive with case and power supply **$695**
New!
Dual Siemens drives with dual case and power supply **$1240**
(90 day limited warranty on disk drives)

master charge VISA

CASES AND POWER SUPPLIES

5½-inch enclosures
Single drive unit case and power supply **$85**
Dual drive unit case and power supply **$120**
(Extender cables are $15 each extra)

8-inch enclosures
Single drive unit case and power supply **$150**
Dual drive unit case and power supply **$250**

Attention dealers, OEM's & distributors
Having trouble finding cases and power supplies for your disk drives? Call us for details on our attractive pricing and immediate product availability.

NEW! TRS—80™ Model III
With two double-density *80 track* disk drives **$2,495**

TRS-80™ SOFTWARE PACKAGES

MAKE80©
Converts 35 or 40 track diskettes into 80 track readable diskettes......... **$14.95**
B-17 tape operating system
For high speed (1700 baud) loading & saving of BASIC/machine language programs, almost 4 times faster than regular TRS-80™ interface **$24.95**

The power behind the drives©

✓39

A.M. ELECTRONICS, INC.
3366 Washtenaw Ave.
Ann Arbor, Michigan 48104
(313) 973-2312

SUPER UTILITY, by Kim Watt.
Stand-alone 24K machine language program for disk includes:

ZAP UTILITY
- Read/modify data regardless of disk protection • One-step track-to-track/sector to sector • Dual cursors; ASCII and Hex. Modify in Hex, Decimal or ASCII • Display disk sectors, display file sectors, copy disk sectors, compare disk sectors, display/modify main memory, search memory or disk for specified string and return location

PURGE UTILITY
- Kill files by file spec or category • Zero out unused directories or sectors
- Compute passwords, change disk's name, date, passwords, protection levels • Directory routine indicates all active and inactive files, their location in directory and status of granules

DISK COPY UTILITY
- Copy *any* TRS-80™ readable disk, regardless of protection

TAPE COPY UTILITY
- Copy *any* TRS-80™ readable tape, regardless of protection or baud rate

DISK REPAIR UTILITY
- *Automatically* repair damaged HIT, GAT or BOOT sectors • Directory check advises of errors • Automatic recovery of killed files • Shows active and inactive files, and their location on the disk

MEMORY UTILITY
- Move memory • Jump to memory • Test memory • Compare memory • Input or output any byte to any port • Zero memory • Exchange memory • Edit memory • Load memory

SUPER UTILITY is now available for **$49.95,** plus $2.50 shipping and handling!

™TRS-80 is a trademark of Tandy Corp.

Visit our retail showroom for a "hands-on" look at our wide selection of TRS-80™ hardware, software and peripherals! Hours: Tues.-Fri. 11-7, Sat. 11-5

The campus hero at the University of Louisville is micro-sized.

The Microprocessor as Tutor

By Ken Reid

Computers are widely used in instruction, and can be employed in very sophisticated ways. Instructional techniques that depend entirely on computer power include real-time simulations of aircraft, industrial processes and the human body (Massachusetts General Hospital simulated patients). But much of today's computer-assisted instruction is less sophisticated. It is usually just a variant of the frame-choice branching program illustrated in Fig. 1.

How small, how simple and how inexpensive can a system be and still provide the logical capabilities needed for this type of instructional program? The answer is, surprisingly, that a microprocessor is quite capable of doing it—not for one student, but for a whole class at the same time. This can be done by carefully distributing the parts of the task among devices specifically designed for their particular function.

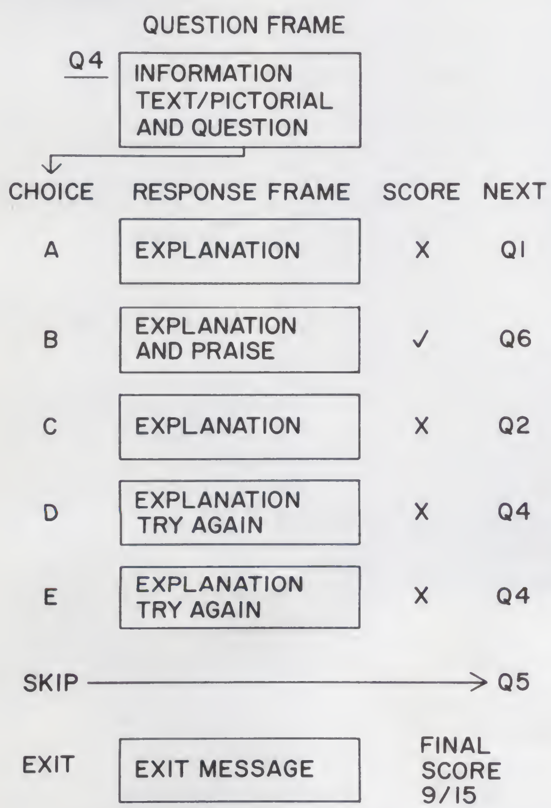


Fig. 1. Logical structure of frame-oriented computer-aided instruction (CAI) or computer-aided evaluation (CAE). (If the text material is sufficient to enable the student to answer the question, it is CAI; if he is being evaluated on information acquired from other sources, it is CAE.) Each question frame offers a choice of possible responses; depending on the response chosen, additional explanatory text is presented. Wrong choices lead to review frames (A,C) or to chances to try other choices for the same question (D,E). The correct choice leads to more advanced material. Some systems permit a student to skip a part he finds irrelevant to his goals; all permit a voluntary exit.

The appropriate parts distribution is illustrated in Fig. 2. Instructional text (including pictures) is stored photographically on microfiche cards, which act as inexpensive, compact and reliable ROM. The logical structure of the lesson is stored separately, either in ROM or on floppy disk, as a 256-byte *linkage list*. This is read into the computer whenever a student requests a new lesson topic. The driver program is content-free; for example, it contains no lesson-specific data, but simply interprets student input according to instructions in the linkage list. The communication with the student is also content-free; the student uses a small seven-key keyboard to respond, and the computer uses a small predefined vocabulary and a three-line video display to inform the student of his progress and to tell him where to look next on the lesson card.

With this system, one microprocessor can maintain 24 active student terminals at the same time, using a polling loop in the driver program. Response time is, at worst, two seconds, and is limited by the need to read linkage list data from floppy disk.

Ken Reid, 1935 Trevilian Way, Louisville, KY 40205.

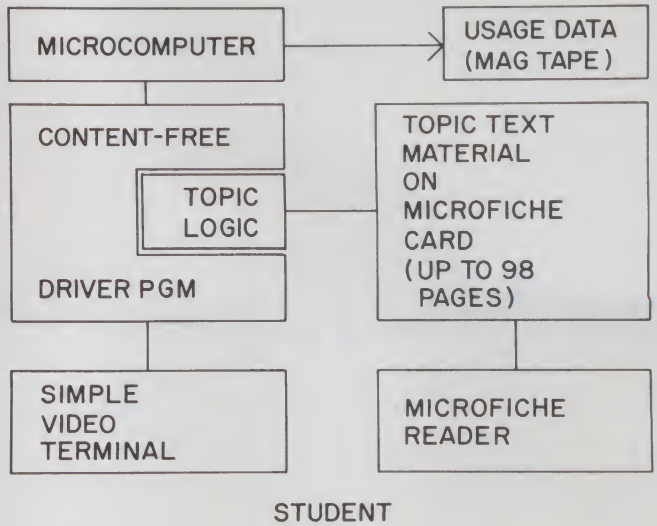


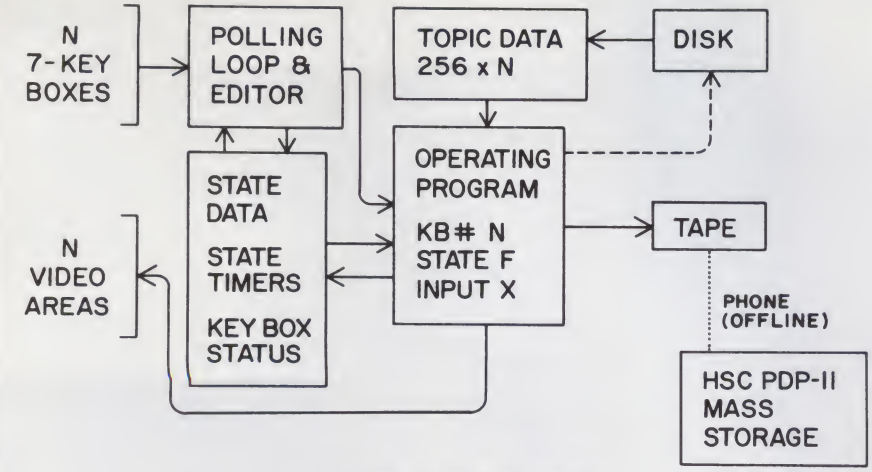
Fig. 2. Logical separation of components required for student-controlled CAI or CAE of the type diagrammed in Fig. 1.

During operation, the driver program (Fig. 3) runs a cyclic scan of all keyboards. When a change in status is detected, the response is read in. Three variables are then passed to the response subroutine: N, the user or station number; F, the functional status of that user; and X, the actual response made. The status variable F is needed because responses have different meanings during sign-on and during the learning interaction. A logical diagram of the F cycle is shown in Fig. 4.

The system so defined is specifically suited to self-instruction and self-evaluation. This is so because the student is the only link between the computer and the manually operated microfiche reader. The reader could be computer-controlled, but its cost would rise from under $300 to over $3000, which is not economically justifiable for this application.

The microprocessor hardware we use at the University of Louisville is shown in Fig. 5. We chose an industrial-grade microprocessor to ensure

The system in use.



CASEGO.N FLOWCHART

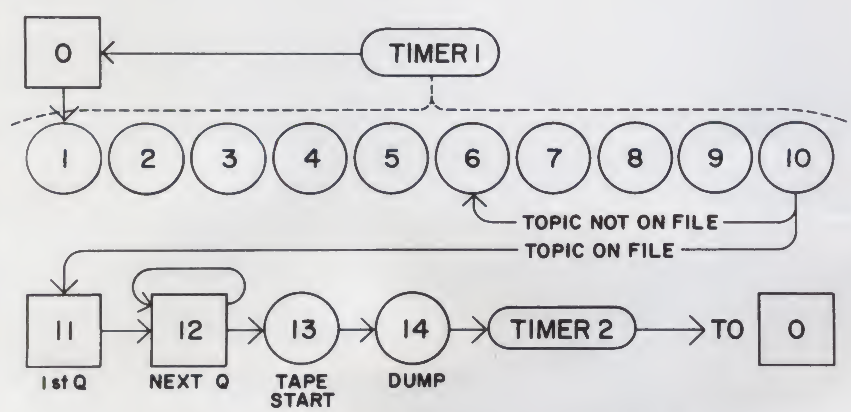
Fig.3. Block diagram of content-free driver program and its interactions with the linkage list (topic data), the keyboxes and the video monitors. The video displays are memory-mapped. Usage data is saved on magnetic tape and transferred off-line to the Health Sciences Center PDP-ll computer disk files.

a high degree of reliability under conditions of continuous use. Our system has now operated for 14 months with less than 5 percent unscheduled down-time; most of our problems have turned out to involve peripheral wiring or program bugs. The system is located in our Health Sciences Library and is available whenever the library is open.

Total hardware cost, including TV monitors, microfiche readers, spare parts and repair equipment, was about $15,000. A single duplicate of the computer, without peripherals, would cost under $5000. The available terminal hours, assuming that the library is open 15 hours a day for 360 days a year, total 129,600, giving a cost-per-available-hour of less than 12 cents.

Not all available hours are used, of course. Our students actually use only about 3000 terminal hours/year, with wide fluctuations in intensity of use depending on course load and the approach of examinations. Cost per hour of actual use is thus about $6.50 to pay off the hardware in one year. Using a more reasonable five-year payoff period would bring hardware cost down to $1.30 per user hour.

Hardware cost is not the whole story, as many CAI users have discovered. Someone must design and write the lessons, and this is high-level, intellectually demanding work. Furthermore, a lesson is not a one-



STATE SEQUENCE DIAGRAM : CASEGO.N

Fig. 4. Transition state diagram for the driver program. During states 1–5, the student enters a five-letter identification code. During states 6–10, he enters a five-letter code identifying the topic he wishes to study. While questions are being answered, the function states 11 and 12 are used; 12 is used for all questions after the first. States 13 and 14 cover activities at the end of the test. States 1–10 time out automatically to state 0 after two minutes to free the work station if the student leaves the terminal. State 14 displays the final score data for four minutes and then reverts to state 0: "terminal free."

The keyboard and terminal.

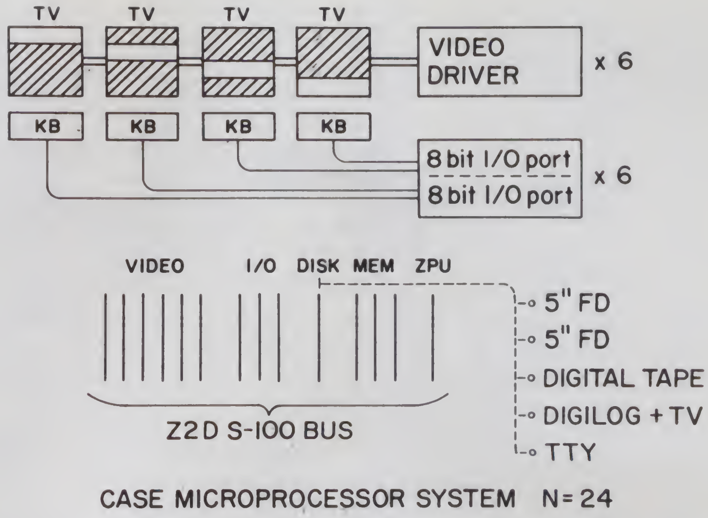


Fig. 5. Hardware configuration for our CAE systems. Six memory-mapped video drivers (SSM VB1, 16 lines × 32 characters/line) are used to drive 24 Sanyo VM4209 video monitors. The monitors are masked so that each user sees only four of the 16 lines. Three of the four lines are used to communicate with the student; the fourth is left blank to minimize problems of vertical drift. Each student has a seven-key input box. The key inputs are converted to three-bit binary code and latched until queried by the polling loop. The microcomputer used is a Cromemco Z2D configured as shown. A digital tape unit (Computer Aid by National Multiplex Company) and a video monitor share one 300 baud serial port; other devices are connected when needed. One five-inch floppy disk is sufficient for operation, but we have two for ease in copying and updating disks.

time investment, but a continuing process, demanding new editions at regular intervals. How do we persuade people to do this work?

Some, indeed many, teachers will do it for fun—at least at first. Much CAI material has been written with no other motivation but curiosity—to try out a new teaching technique. But the novelty wears thin after the second or third revision.

Some large CAI networks, notably the Plato project of the University of Illinois, have hired Ph.D.s fulltime as writers. This gets the work done, but few schools can afford this route.

There is, however, a well-known middle road, familiar to writers of textbooks. This is the payment of royalties according to use.

How do you measure use? The microcomputer does this as part of its normal operation. As a student interacts with his lesson, each question and response is recorded. When he finishes a card, this usage history is dumped to cassette tape. Periodically the tape is picked up and read, over the telephone, to a disk file on a large computer. At the end of the year, this file is processed, and a report is prepared on each lesson topic. The report is sent to each author for use in preparing next year's edition of the lesson topic.

The overall circulation of information in the system is summarized in Fig.6.

Does it work? Yes. Table 1 summarizes use over the past five years. Until January 1979, we used terminals on a large time-shared computer. We switched to the microprocessor because the large system was becoming overloaded with other uses and the response time was getting too long. Since the changeover, use has increased. Our main problem now is persuading authors to keep topics up-to-date and to improve the quality of their material—problems that we share with publishers of text material in the traditional paper and ink format. ■

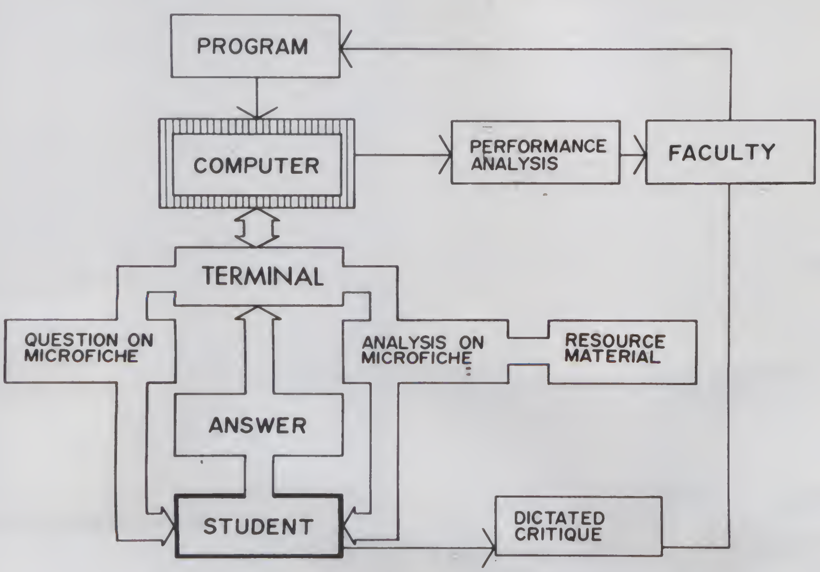


Fig. 6. Conceptual layout of our current systems for computer-assisted self-evaluation. In addition to the operating program (Fig. 3), other programs on a larger computer are used to edit and check instructional material, code linkage lists and analyze usage data. Student comments are telephoned to the office or provided in written form through a suggestion box and an annual class survey.

Year	Spring	Fall
1979	73,003	25,829
1978	45,231	21,088
1977	60,535	20,873
1976	65,279	15,000
1975	61,282	N.A.

Table 1. Usage data in responses per semester. In 1975 we operated only during the spring semester. The transfer to the microprocessor occurred on January 15, 1979. The upsurge in use compared to spring 1978 is attributed to better response time. The educational material available for use in fall semester courses is still quite limited in comparison to that available for spring semester courses.

SUPER BRAIN QD™

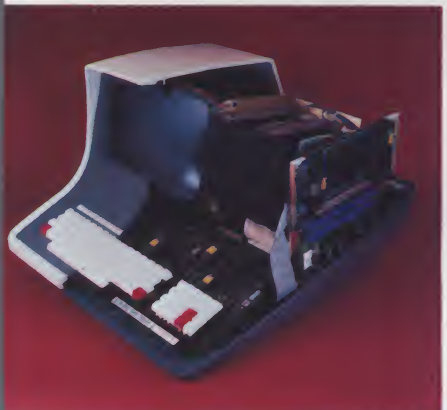
Once in a great while someone comes along ith a simple improvement for an already great roduct. Take our SuperBrain, for example. Really a imple concept. A high-powered, low cost micro- omputer packaged in an attractive desk top abinet. So how do you improve on that?

/E DID IT...

It wasn't enough that our SuperBrain had uch standard features as twin double density ¼" drives with over 300,000 bytes of disk torage. A full 32K of dynamic RAM - expandable 64K in seconds. A CP/M* Disk Operating ystem which assures compatibility to literally undreds of application packages presently availa- le. A crisp, 12" non-glare screen with a full 24 ne by 80 column display. A full ASCII keyboard ith a separate keypad and individual cursor ontrol keys. Twin RS232C serial ports for fast *nd easy connection to a modem and/*or a printer. nd, dual Z80 processors which operate at 4 MHZ insure lightning-fast program execution. No, it asn't enough. So we made it better.

NNOUNCING SUPERBRAIN QD...

Our new QD model has all of the features of ır phenomenally popular SuperBrain with the ldition of double-sided disk drives and an extra 2K of dynamic RAM. So, for only a modest in- ease in price, you can order your next SuperBrain ith more than twice the disk and memory storage apability. But, best of all, the new QD model has e same tough, rugged construction and ceptional quality that made our SuperBrain ıch a success.



HOW DID WE DO IT?

The secret of SuperBrain QD's incredible disk storage lies within our new double-density double-sided disk drives. A total of nearly 720,000 bytes of data are formatted on two specially designed 5¼" drives. And that's more than enough to get you started with most serious small business applications. And SuperBrain QD's standard 64K of dynamic RAM will handle even the most complicated programming tasks.

Of course, if you're into megabytes instead of kilobytes, you may think neither SuperBrain is right for you. Not so! Intertec offers 20-96 megabytes of hard-disk storage which connects in seconds to either the SuperBrain or SuperBrain QD. So, your original investment is always protected. As you grow. No matter how much your needs expand.

BUT IS IT RELIABLE?

Our best salesmen are our present users. Not only have SuperBrain users been impressed with the inherent reliability of the system, they tell us that no other microcomputer system available today offers such a unique modular design concept. Just about the only tool required to easily maintain the system is a common screwdriver. And Intertec's total commitment to product service and customer support, with service outlets in most major cities, insures your original investment will be a valuable one for many years to come.

THE DECISION IS YOURS.

Whether your next SuperBrain is a regular model or our QD version, you will have the satisfaction of knowing you purchased what is becoming one of the world's most popular microcomputer systems. And regardless of which model you choose, you'll probably never outgrow it because you can keep expanding it.

So, call or write us today for more information. Intertec systems are distributed worldwide and may be available in your area now.

INTERTEC DATA SYSTEMS®

2300 Broad River Rd., Columbia, SC 29210
(803) 798-9100 TWX: 810-666-2115

196

gistered trademark of Digital Research, Inc.

6801: A One-Chip System

By H. W. Neff

Considering the current 16-bit brouhaha, the idea of a new eight-bit microprocessor is almost passe. But the Motorola 6801 is definitely not ho-hum. In terms of software, this "new" machine is an old friend with some significant additions to its bag of tricks. And hardware? Just add power supply, crystal and your own abilities.

In the last couple of years electronics literature has chronicled Motorola's additions to the M6800 family of eight-bit microprocessors. While maintaining the basic architecture of the original machine, the evolution of these newer processors has proceeded on two fronts: performance and functionality. Fig. 1 illustrates the relationship between the 6800 and its descendants.

Motorola doubled the speed of the original machine with the 68B00. The next boost came with the 6809. Its overall performance has been hiked by three or four times that of the original 6800, making it a 16-bit wolf in eight-bit sheep's clothing.

The company has been cramming more and more onto less and less silicon. This first produced the 6802, a 6800 processor plus a clock (6875) and 128 bytes of RAM (6810). Like the 6800, the 6802 packs the punch of 80 pins' worth of processor and support into a 40-pin package, though the two are not pin-for-pin compatible.

The next device to evolve is Motorola's MC6801, which is further above the 6802 than the 6802 is above the 6800. Because of the HMOS design and fabrication process, it is smaller, faster and has lower power.

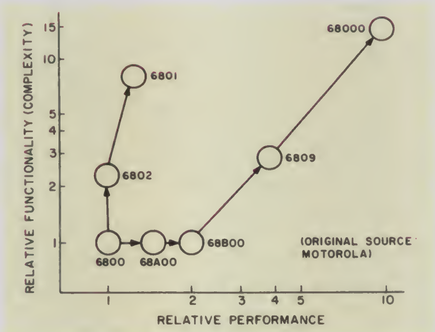


Fig. 1. Evolution of the 6800 microprocessor family.

The phenomenal hardware density of the 6801 is the key to its increased functionality, while several new instructions and faster instruction timing give it a boost in the performance range as well. This is reflected in the relative position of the 6801 in Fig. 1.

Hardware Density

The 6801 represents a significant advance in hardware density over the original 6800. Both devices come in 40-pin packages, but the 6801 is roughly equivalent to the original 6800 plus six family support devices. The equivalent pin count for the 6801 is in excess of 180 6800 era pins. Fig. 2 illustrates the hardware relationships between the 6800 and several of its descendants, including the 6801.

The 6801 also has an on-chip clock oscillator, comparable to the 6875. Only an external crystal at four times the desired clock rate is required for machine timing. For multiprocessor operations, an external 4× single-phase clock may be provided in place of the crystal. Originally, a 6801E was planned for external clock operation, but the 6801 is capable of operating in both modes.

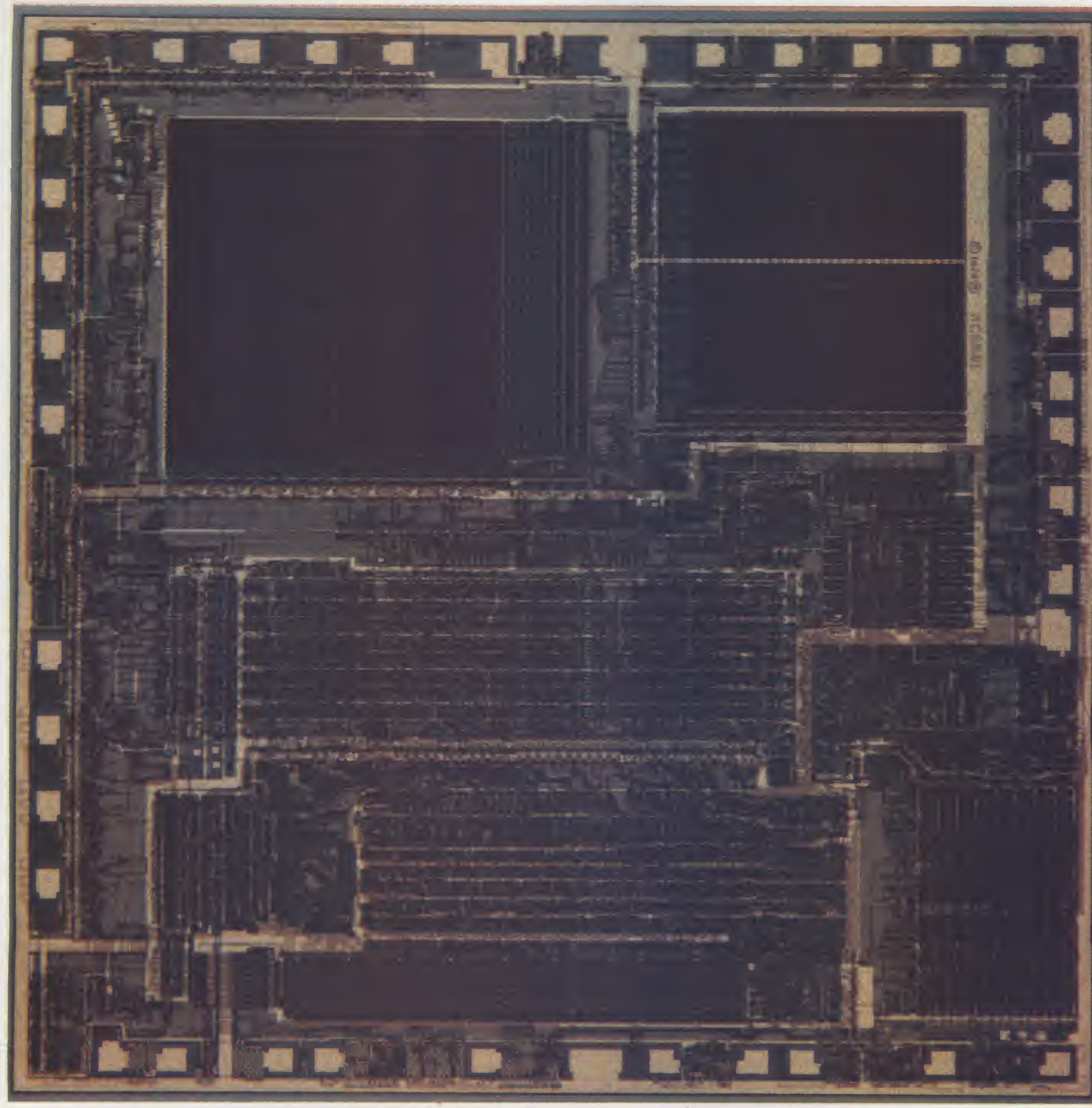
The 6801 provides an on-chip serial communications interface (SCI) comparable to the 6850 ACIA. The SCI is controlled by four registers and is capable of several operational configurations. Two different serial data formats (NRZ and bi-phase) are provided at any of our different internally derived data rates. Or, if required, you may configure the SCI to operate with an external clock source.

The SCI also sports a "wake-up" feature, which is useful in multiprocessor communications where the first information is the address of the destination processor. After monitoring enough of the message to determine the destination, the nonselected processors can ignore the remainder of the message by putting the SCI to sleep. The SCI will automatically wake up upon receipt of ten consecutive 1's, which the protocol should define as the line idle condition between messages.

The 6801 contains a 16-bit programmable timer that is functionally equivalent to a third of the 6840 support chip. The 6801's timer is a three-function resource. The timer has a free-running 16-bit counter, which may be read at any time by software. The timer is capable of generating output pulses with a programmed width and simultaneously measuring the width of an independent input pulse.

The 6801 has extensive on-chip

H. W. Neff, P.O. Box 3359, San Leandro, CA 94578.



The 6801 chip. (Photo courtesy Motorola Semiconductor Products, Inc.)

memory resources. It has a 2K ROM (equivalent to the 68316) and 128 bytes of RAM. The RAM is functionally equivalent to the 6810, with the additional ability to maintain 64 bytes when the 6801 is in power down mode. The on-chip ROM is mask-programmed at the factory with the pattern provided by the buyer (along with an order for approximately 1000 pieces).

For evaluation purposes, Motorola makes a 6801 with a 2K monitor called LILBUG already in ROM. Details on the LILBUG monitor and a source listing are available from Motorola. The 6801 with LILBUG is identified as the MC6801L1 and costs approximately $50. For prototyping and small-quantity jobs an EPROM version (the 68701) and a ROM-less version (the 6803) are available. The prices for these devices are approximately $150 and $45, respectively.

New Instructions

The 6801 is source- and object-code-compatible with the 6800. In addition to the original instructions of the 6800, 12 new instructions are available. These include the much-needed push and pull stack operations for the X (index) register and seven 16-bit operations. These instructions operate on the newly defined D register.

As shown in Fig. 3, the D register is formed by the concatenation of the A and B registers. Included in the 16-bit operations is an 8×8 multiply. The multiply instruction uses the contents of A and B as multiplicands and

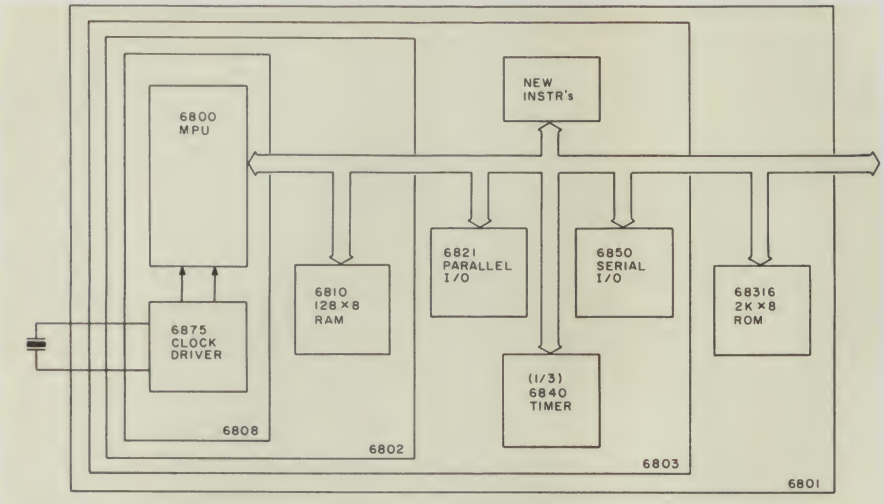


Fig. 2. Functional hardware comparison of 6800 family processors.

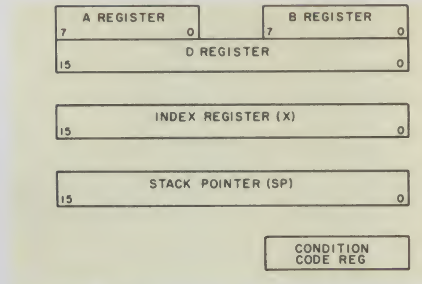


Fig. 3. Programming model for the 6801.

stores the product in D.

Two instructions have been modified in the 6801. The JSR (jump to subroutine) instruction has been altered to include the direct addressing mode. The second changed instruction is CPX (compare index register). The CPX instruction has been modified to appropriately set or reset additional flags in the condition code register. This may have subtle side effects in older code transported to the new machine, so keep a watchful eye.

The new and modified instructions for the 6801 are summarized in Table 1. For more details, refer to the 6801 data sheet and programming reference card. Both items are available at no charge from Motorola.

Faster Instruction Execution

The addition of the new instructions gives the 6801 increased capabilities over the 6800 and 6802. The 6801 also gets a performance boost from some improved instruction execution times. All of the Jump and Branch instructions have been made one cycle faster. This amounts to a 25 percent decrease in execution time for these instructions. In the indexed addressing mode the execution time of instructions has been shortened by one cycle. For most instructions this results in a 20 percent reduction in execution time.

Note that one instruction has actually become slower in the 6801. The increased functionality gained in the modification of the CPX instruction has made it one cycle slower than before. In the indexed addressing mode this extra cycle is offset by the increased speed of the mode for all instructions, resulting in an unchanged execution speed.

One Chip Minimum System

The minimum system for a 6801 consists of a 6801 and from zero to three support chips. The reason for the variance lies with the 6801 itself. On reset the 6801 is hardware-programmed into one of three basic configurations. The configuration requiring no additional support is rightfully dubbed the single chip mode. In this mode the 6801 looks like four parallel ports and a serial port to the outside world. There is no external address or data bus available.

For systems requiring only a few bytes of I/O or external memory, the 6801 is configured into its expanded nonmultiplexed mode. In this mode two of the eight-bit parallel I/O ports become the external data and address buses. The external address bus is only eight bits wide, thus limiting the external memory or I/O to 256 bytes or less.

When more external resources must be accessed, the 6801 must be configured into its third operating mode. This mode (for those not tipped off by the name of the last one) is the expanded multiplexed mode. The expanded multiplexed mode uses two parallel ports as a high-order address bus and a multiplexed address and data bus. This allows all 16 address lines to be brought out, giving a full 64K address space. The address and data lines can be demultiplexed simply by the addition of a 74LS373 (or equivalent) octal latch, as illustrated in Fig. 4.

The expanded multiplexed mode can be particularly cost-effective for small systems. A $45 6803 (a ROM-less 6801) plus a $20 EPROM and $5 worth of TTL is a lot cheaper than the $150 price tag on the 68701 (EPROM version).

If the prototyped system is to be duplicated in sufficient quantity to justify the ROM costs, there can be additional pleasant surprises inherent in the use of a 6801. In the expanded multiplexed mode you can configure the 6801 to place the internal ROM at F800 to FFFF. This is the

Index Register (X) Instructions	
ABX	Unsigned addition of the B register to X.
PSHX	Push X onto stack (least significant byte first).
PULX	Pull X from stack (most significant byte first).
CPX	Compare X. Modified for increased condition code register effects.
Program Control Instructions	
BRN	Branch never. Equivalent to a 2 byte NOP.
JSR	Jump to Subroutine. Direct addressing mode has been added.
16 Bit Instructions	
ADDD	Addition of memory to D without carry. Results are left in D.
ASLD	Shift D 1 bit towards MSB. MSB shifted into carry. LSB is cleared.
LDD	Load D from memory.
LSRD	Shift D 1 bit towards LSB. LSB shifted into carry. MSB is cleared.
MUL	8 × 8 multiply of A and B. 16 bit result left in D.
STD	Store D in memory.
SUBD	Subtract memory from D. Results are left in D.

Table 1. New and modified instructions for the 6801.

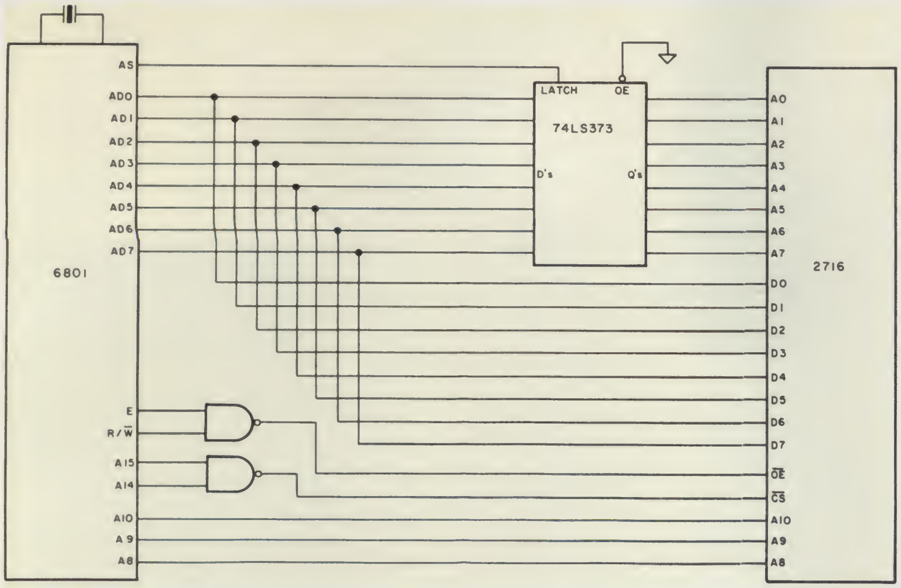


Fig. 4. 6801 expanded multiplexed mode minimum system with external EPROM.

same 2K area of memory occupied by the external EPROM illustrated in Fig. 4.

This leads directly to the pleasant fact that no new software needs to be developed when putting the code into ROM. (The less significant cost of the external EPROM and TTL also disappears.)

And finally, when the time comes to prototype the next system, just configure a 6803 out of one of the (inevitably produced) spare 6801s from the previous system and begin. This is a no-waste versatility virtually unmatched by any other processor available.

The 6801 was officially unveiled more than a year ago but is only now becoming widely available. The new instructions, timing and hardware density result in a powerful and easy-to-use device that is the latest development in the art of putting a computer system on a chip. The 6801 is ripe for plucking by anyone with the slightest motivation, and I fully expect to see it appearing on several boards for home systems in the coming months.

Data sheets and other 6801 literature may be obtained by contacting Motorola at: Technical Information Center, Motorola Semiconductor Products, Inc., P.O. Box 20924, Phoenix, AZ 85036.■

On the Road With a TRS-80

By Ronald H. Bobo

A van rolls up to a midwestern business establishment. A tall, bespectacled young man with a beard emerges, runs a 100-foot extension cord to the nearest power source. Computervan is plugged in, ready to provide the business with on-site custom programming.

Computervan is just one feature of COMP-U-TRS, a unique new store in the growing St. Louis suburb of Florissant, MO. Sitting right next to a Radio Shack Computer Center, COMP-U-TRS sells software, and software only. Founder and owner John Knoderer claims that it is the first retail software store of its kind in the country.

Certainly, the van itself would qualify Knoderer as a free-thinking entrepreneur. The custom-designed vehicle roams the countryside from St. Louis, MO, to Leavenworth, KS, providing customers with programming and consulting services while their own computers remain free for normal use. What could be more convenient?

Considering Knoderer's background, it's not surprising that he comes up with such original ideas.

A member of the high-I.Q. society Mensa, Knoderer has always been interested in math. This, he feels, was probably responsible for his later affinity with computers.

"In the third grade," John recalls, "the teacher came up with what I realize now was probably meant as an impossible question, with no expectation of an answer coming from the students. The question: What is one divided by two? I came up with 1/2, but don't ask me how."

By sixth grade, Knoderer was placed in a special advanced class. These classes continued through high school, getting him into college calculus before graduation.

While Knoderer was in high school, his father was attending Indiana University. The lab where he worked just happened to be next door to the computer center. Knoderer would accompany him to campus, then hang around the computer room, going through all the thrown-away punch cards, sorting them by color and whether or not they were plain or punched. Then, in his senior year, Knoderer joined a computer club and gained some hands-on experience.

His first course in college was computer programming (in FORTRAN). This gave him a user number for the college computer, a CDC 6600. On Friday nights when most people were out on dates, Knoderer would be in the computer room until early morning, writing and running programs, experimenting and learning as much as possible.

Ronald H. Bobo, 3246 Gravois, St. Louis, MO 63118.

Besides FORTRAN, he took courses in COBOL and Systems Analysis by Computer.

"Because I was a business major," he says, "I think I have learned more about computers than I ever would have otherwise, because there is so much you can do with a computer in business if you know the business aspects. A lot of programmers do not understand the field of business, nor do most people in business understand programming."

Front of COMP-U-TRS, conveniently located next to a Radio Shack Computer Center.

Into the Classroom

After college came a job as an elementary school teacher in Leavenworth, KS. Not surprisingly, Knoderer used a computer in his classrooms.

He ordered a TRS-80 the same day he found out it existed, in August 1977. It was finally delivered in November. In the meantime, though, open house was coming up at the school in October, and Knoderer thought it would be wonderful if his computer could be on display.

He wrote a letter to Lewis Kornfeld, Radio Shack's president. Knoderer's computer didn't arrive in time, but Kornfeld arranged to have the regional manager supply a demonstrator unit. They flew it in from Denver, and Knoderer had it for three days.

Part of the left wall at COMP-U-TRS, showing software offerings from Instant Software and other publishers.

Knoderer wrote CAI (computer-aided instruction) programs for the TRS-80 during his two years of teaching. Don't be surprised to see them show up on the market one of these days.

During his second year, Knoderer decided that the pressures of teaching were not for him. When the principal, who had originally hired him because of his computer background, said he didn't think the school board was going to approve a permanent computer project, Knoderer gave notice that he would not be back the following term.

Meanwhile, Knoderer had been attending meetings of a TRS-80 users' group in Kansas City. Gradually, members began to seek him out for help with programming problems. A friend suggested that he should be getting paid for these services, and the idea for the consulting and programming service was born.

Knoderer left his teaching job in April 1978 and started his travels, his TRS-80 riding along in the back of a Vega. By May he had enough commitments from customers to justify ordering a custom-designed van. It arrived in June, and Computervan, Inc., was officially launched.

COMP-U-TRS Takes Off

COMP-U-TRS started up in August. At first, the store sold software only for the TRS-80s. Knoderer had just received his Model II, probably the first one sold in the St. Louis area, and was happily writing software for it. But COMP-U-TRS hadn't been open long before people were requesting programs for the Apple and PET.

Knoderer deliberately chose the location next to the Radio Shack Computer Center. This was a wise decision, since there has been a remarkable paucity of software from the Shack.

In spite of some early apprehension from the Computer Center people, this proximity probably helps both businesses. Knoderer refuses to sell computers, so all potential computer buyers are referred next door. Conversely, when Radio Shack has recommended a particular computer model and the customer approaches COMP-U-TRS for software information, John's employees are under orders not to suggest another model.

Knoderer fears, however, that the name may have to be changed. He has been getting some static from Radio Shack, and they've refused to accept his listing in their forthcoming roster of independent software suppliers because of the TRS in the store's name.

In Knoderer's opinion, he has probably generated around $20,000 worth of business for Radio Shack by being in his present location.

Needless to say, several TRS-80 Model I's are on hand, as well as a Model II. Knoderer has recently installed an Apple II, and a PET is available part-time for program demonstrations.

Company policy states that a po-

tential customer is free to try any software before buying, using one of the available systems. Systems may also be rented for such in-store uses as game playing, for a nominal charge of $2 per hour and $1 for each program used. The money may then be applied toward a software purchase.

The in-house stock includes software from all the major publishers and many offerings from smaller companies. The store carries a sizable selection for Apple and PET. In addition, a large selection of computer magazines and books is on hand.

Knoderer sells some disk drives and printers, but this is the extent of the hardware. He says that if he were to sell computers, or take commissions for selling them for others, he might become biased toward one type of machine. Besides, he feels there is more money to be made in the software and consulting end of the business.

Not All Roses

What are some of the problems?

For one thing, people sometimes want to use the systems, but don't buy anything. Hopefully, the hourly charge will discourage this.

"And," he says, "there's the person who buys only a blank tape before using a machine. I then have to wonder if he's saving programs.

John Knoderer, owner of Computervan and COMP-U-TRS, at the custom programming position in Computervan. Note disk drive at left and Base 2 printer on shelf. Shelf was actually designed to accommodate a Line Printer I.

Probably I'll have to rearrange the leads on the machines so it won't be possible to save a program to tape."

Another problem is that the business was slow to show a profit. The Computervan operation subsidized the store during the first several months of operation.

Any other drawbacks?

"It takes some serious thinking to decide whether to exhibit at a show. Take the Amateur Radio and Computer Hobbyist convention held in St. Louis recently. To really do it up good, I took all my employees. We sold about $2000 worth of merchandise, but it cost $2500 to attend! On the other hand, a lot of people now know we're here who didn't before."

Looking Ahead

Once the store becomes wildly successful, will John give up his mobile programming service?

"No," he says. "I want the store to become successful enough so that Ben, my manager, is making more money than he knows what to do with. This will give me the freedom for more programming. Also, there are several more aspects of the computer business in which I may become involved. Right now I have a friend who wants to start a wholesaling operation and I'm looking into that."

John is also making plans for his next van, which will be larger (something like a motor home, I suspect). Plans call for this one to have a self-contained power unit so John can be programming, if he wants, while someone else drives.

What will the recession do to the computer business?

John thinks it may hurt the sale of new computers somewhat, but computer owners will still be buying programs.

"I haven't noticed any decrease in the number of customers so far, and they all seem to have money. Credit restrictions may slow down hardware sales. One of my customers recently wanted to obtain a Spinwriter but couldn't get a lease approved."

Drawbacks notwithstanding, it appears that the potential for profit should be great in this type of store, as long as some judgment is used so that markets don't become saturated. Any fairly large city should be able to support at least one establishment similar in concept to COMP-U-TRS.

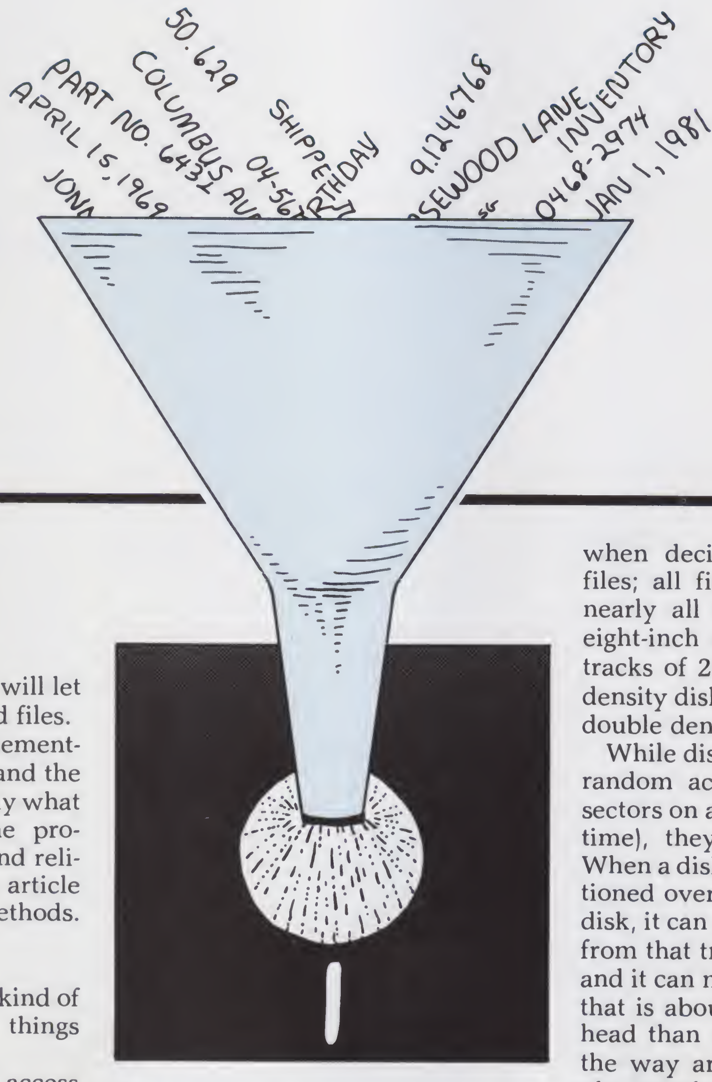
Everyone, of course, will not locate next door to a Radio Shack. Locating somewhere other than a shopping center would keep overhead down, and perhaps a few judiciously placed ads could bring in a fair amount of mail order business.

John Knoderer has already proved that money can be made by the experienced TRS-80 programmer, as evidenced by the success of Computervan, Inc. Programming and consulting, of course, need not be confined to the TRS-80. If your expertise lies elsewhere, the same business could undoubtedly be implemented using any other widely sold microcomputer.

These two types of businesses would probably be successful in many parts of the country. Who will be the next entrepreneurs?■

A look at the three commonly used methods of disk space management.

Data on Disk: Implementing File Systems



By Richard Fritzson

Any disk operating system will let you create and use named files.

However, files can be implemented in several different ways, and the method used can affect not only what features are available to the programmer, but also how fast and reliable the disk storage is. This article looks at the details of three methods.

Disks

Every year, it seems, a new kind of disk is marketed. But some things about disks never change.

Each has a smallest unit of access called a sector; you cannot read or write anything smaller than one sector at a time. Sectors vary in size from 128 bytes to more than 1024; one system uses a sector size of 3584 bytes. The sectors are arranged on concentric tracks on the surface of the disk. The number of sectors on each track depends on the size of the disk and, of course, on the size of the sector.

Neither the sector size nor the total disk size makes much difference when deciding how to implement files; all file systems will work on nearly all disk types. The popular eight-inch soft-sectored disk has 77 tracks of 26 sectors each. On single density disks a sector is 128 bytes; on double density disks it's 256 bytes.

When a programmer asks you (the system) to create a file, how do you decide which sectors of the disk to give him?

While disks are often referred to as random access devices (that is, all sectors on a disk can be read in equal time), they are only semi-random. When a disk drive's read head is positioned over a particular track on the disk, it can more quickly read sectors from that track than from any other, and it can more quickly read a sector that is about to pass under the read head than one that has to travel all the way around the disk. This fact plays a role in determining how to allocate files.

Files

Disk files give the programmer a chance to associate a symbolic name (such as LEDGER or STARTREK) with a part of the disk. Each file consists of a series of records of a fixed

Richard Fritzson, 25 Callodine Ave., Amherst, NY 14226.

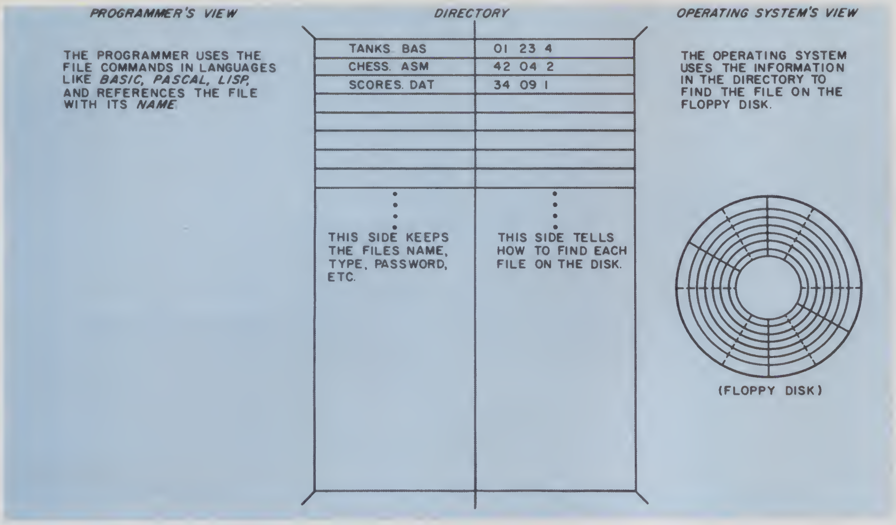


Fig. 1. The directory links the programmer's view of files to the actual floppy disk.

size, usually the same as the disk's sector size. (Note that variable record sizes found in some BASICs are provided by BASIC and not by the disk operating system.) All systems let the programmer create and delete files and read and write data sequentially, as though the file were a cassette tape.

Beyond these basics, some systems provide other important features. Foremost among these is random access to the data in a file. Since part of a disk's power lies in the fact that it does let you read any sector independently of all the rest, a system that restricts access to its files to sequential reading and writing is doing a great disservice to the user. No reasonable form of database can be built without random access files.

Another useful feature offered by some systems is the ability to dynamically expand a file; that is, to take a file that has already been created, open it and add new records to the end of it, without having had to reserve space for them when the file was first created. As with random access files, not all methods of implementing files permit this feature.

Also, some systems allow file names to be 11 characters long, some only eight characters, and some only six. Some systems allow files to be marked as read-only, preventing accidental or malicious damage to data. Some implement password-protected files. Most of these amenities can be added to a system no matter how the file structure is implemented.

The Directory

Before describing the different ways of implementing disk files, we should take one look at something common to all of them—the disk file directory (see Fig. 1).

This data structure resides on each disk and can be logically divided into two parts: one keeps track of how many files there are and their names (along with their types, passwords, etc.), while the other keeps track of where each file actually is on the disk. Naturally, exactly what information is stored in the second part depends on how the files are implemented. The method of describing the location of each file's sectors comes after you have decided how to solve the main problem of managing files: when a programmer asks you (the system) to create a file, how do you decide which sectors of the disk to give him?

There are three commonly used solutions to this problem: sequential allocation of disk space, linked allocation of disk space and mapped allocation of disk space.

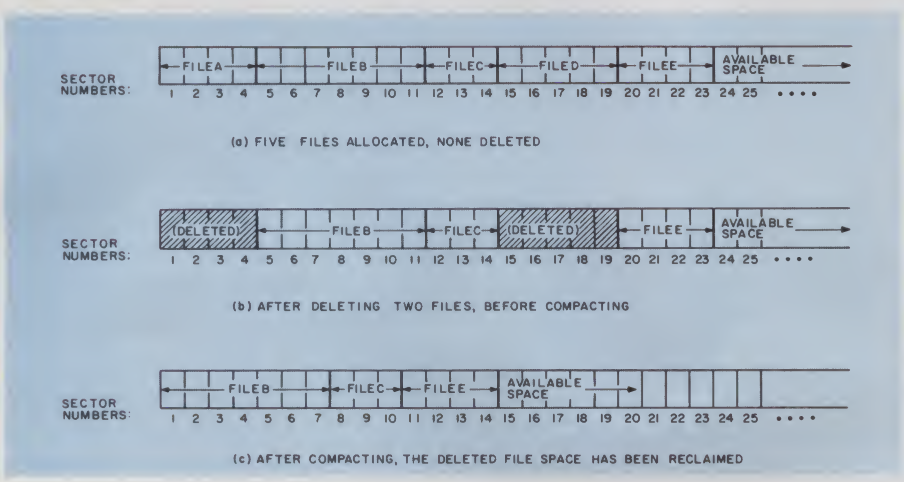


Fig. 2. Sequential allocation of disk space.

Sequential Allocation

Sequential allocation is the easiest method of disk space management to implement. All you do is number each sector on the disk from 1 to N (depending on the disk size). To create a file, you have to specify how big it is going to be. The operating system allocates the requested number of sectors beginning with the first empty one and continuing, sequentially, for as many as needed. Once a few files have been created, the disk space looks something like Fig. 2a.

The directory for this type of system usually contains the number of the first sector of the file and the number of sectors allocated for the file (Fig. 3). The system can find the next available sector by looking at the entry for the last file.

Besides its simplicity, this method has other advantages. Given the semi-random nature of disk access mentioned earlier, a system that keeps all of the sectors of a file close together will provide faster access to the data in the file than one that does not. Sequential allocation always keeps all of the sectors of a file as close together as possible. This produces very fast disk I/O.

Furthermore, because the directory contains all the information needed to find any requested record of a file (the first record is listed there; to find the second one just add one to the number listed, to find the third record add two, and so on), files allocated this way can be accessed randomly. Any sector of a file can be read or written without having to first read all earlier sectors.

However, the simple method is not always best. Sequential allocation has some serious defects. For one

FILE NAME	FIRST SECTOR	FILE SIZE
FILEA	1	4
FILEB	5	7
FILEC	12	3
FILED	15	5
FILEE	20	4

Fig. 3. Typical file directory for sequential allocation method.

thing, it is impossible to dynamically expand a file. Fig. 2a shows that there is no way you can add records to the end of FILEA—at least, not without reading and rewriting every record of files B, C, D and E.

A more serious problem shows up when trying to reclaim available disk space. After the creation and deletion of several files, a disk will look something like Fig. 2b. While a clever system may be able to use the deleted file areas for the creation of small files, in the end there is nothing to do except squeeze out the deleted spaces by moving every file down toward the beginning of the disk, leaving all of the empty space at the end of the disk (Fig. 2c). This process is called compaction or squashing. It is a major operation, since it involves reading and writing every record of every file while moving the disk's head back and forth over the surface of the disk. On large Winchester disks it takes a long time. It increases wear and tear on both the individual disk and the disk drive.

If a disk or drive is in marginal condition, this process will often turn up an error. Because the disk is in an intermediate state during the process, with half the space compacted and half not—even half a file moved and half not—this is one of the worst times to get a disk error. Systems with two disk drives can reduce the danger by compacting from one disk to another, but that only partially reduces the wear and tear on the system.

Still, the simplicity of this method makes it popular. The North Star DOS allocates its files in this way; so do the UCSD-Pascal and Ohio Scientific operating systems.

Linked Allocation

Attempting to remedy the problem of reallocating unused disk space has led some to linked allocation of disk space. In this method, the sectors of the disk are not viewed as an array of sectors, but rather as a linked list of sectors. This means that each sector contains a number or pointer that is the number of the next logical sector in the file. This number is sometimes, but not necessarily, the next physical sector on the disk (Fig. 4a).

A file in this system is a linked list of sectors ending with a sector with a special "end of file" pointer. All of the unused sectors are kept in one linked list. Whenever a new file is to be created, its sectors are taken from the list of unused sectors. With this type of allocation, the directory entry for each file contains the first sector of the file, and often the last sector as well (Fig. 5). (Even though the last sector can be found by following the links from the first one, it is often convenient to be able to find it quickly. Also, some systems link the sectors of a file both forwards and backwards.)

This method takes care of the two problems of sequential allocation. Reusing the space from a deleted file is now easy; you just attach the list of sectors from the deleted file to the list of unused sectors by pointing the end of the unused sector list to the beginning of the deleted file. Eventually, the sectors will be reallocated; there's never any need for compacting the disk.

Furthermore, it is now easy to dynamically expand a file: you just get some unused sectors from the free list and hook them to the end of the file. You can, if you want, dynamically shrink files and reuse those sectors as well.

But—and this is a very big BUT—

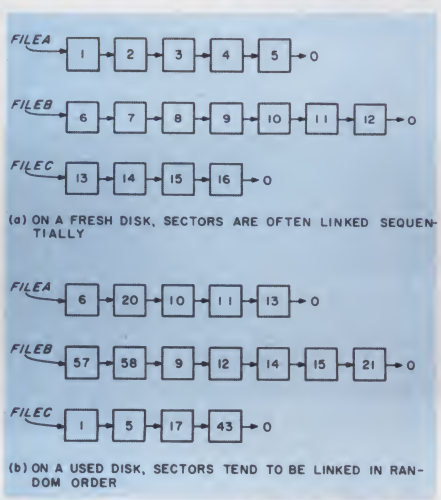


Fig. 4. Linked allocation of disk sectors.

FILE NAME	FIRST SECTOR	LAST SECTOR
FILEA	6	13
FILEB	57	21
FILEC	1	43

Fig. 5. Typical file directory for linked allocation method.

linked allocation introduces problems of its own. After creating and deleting files for a while, the order in which sectors are allocated is not at all related to their actual order on the disk (Fig. 4b). This means that while simply reading sequentially through a file, the disk head will move back and forth all over the disk, slowing down access to the data and increasing wear on the disk and drive.

Worse, with linked allocation, there is no way to perform random access to the records of a file. The only way to find the tenth sector of a file, for example, is to find the pointer from the ninth sector, the only way to find the ninth sector is from the eighth and so on. You always have to read through a file to find a particular record. Not even the systems that link the file both forwards and backwards help solve this problem.

Because of this, linked allocation is not used that often.

Mapped Allocation

The method of disk space allocation that comes closest to ideal so far is mapped allocation. Even though it is used by some very popular operating systems (e.g., Digital Research's CP/M and the derivative systems such as Cromemco's CDOS), it is not widely understood. It's a little more complicated than the other methods.

In its simplest form, mapped allocation means that the directory contains, for each file, the sector number of every sector in the file (Fig. 6). This array of sector numbers is a map of the file; hence the name of the method. To read the fifth record of a file, for example, the system looks at the fifth number in the map for the file and reads that sector.

Unlike the other methods, the mapped allocation scheme does not maintain on the disk some indication

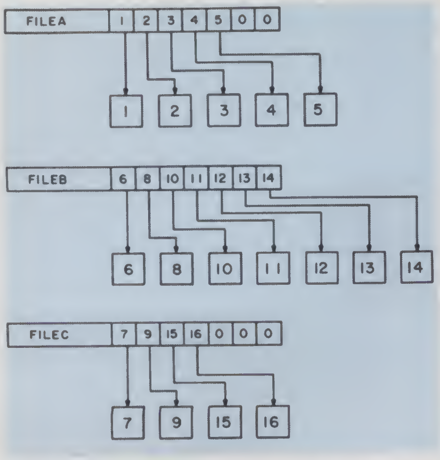


Fig. 6. Simple mapped allocation scheme. Each directory entry points to the sectors of the file.

of which sectors are unused and available for new files. Instead, each time the system sees a new disk, it goes through a logging in or registering process. This involves reading the entire directory and building a bit map of the disk's sectors (Table 1).

A bit map is an array of single bits, each one of which corresponds to a sector on the disk. If a sector is in use in some file, its corresponding bit is set to one; unused sectors have their corresponding bits left as zero. The bit map (sometimes called an allocation vector) is then used to find and allocate available disk sectors.

Like the linked allocation scheme, mapped allocation systems can dynamically reuse deleted file space. Whenever a file is erased, the system reads its directory entry and resets the bits corresponding to each of its sectors to zero, thus making them available for use. No compaction process is ever needed. And, like linked allocation, mapped allocation allows for dynamic expansion of files. New blocks can be found in the bit map and added to a file's directory entry when needed.

Unlike linked allocation, this method allows random access to disk files. Since all of the records of a file are listed in the directory entry, any one of them can be found instantly. And while the sectors of a file can wind up spread out on the surface of the disk (as in the linked method), this problem is minimized by the use of the bit map to allocate available sectors that are adjacent to or near one another. (This information is not easily available from the linked list of available sectors in the linked allocation method.)

Mapped allocation would seem to be the ideal way to implement files except for one thing: it's not quite as simple as I've described.

First, since a file can have thousands of records, a directory entry which listed out the address of each sector would be much too large to be acceptable. For example, a file with 128 records, assuming 16-bit addresses, would require a directory entry of 256 bytes for just the allocation information.

The solution to this problem is to allocate the disk's sectors not one at a time, but a bunch at a time. That is, instead of assigning a number to every consecutive sector on the disk, number consecutive blocks of, say, eight sectors. Now each number in a directory entry's map stands for a block of eight sectors.

You don't have to read and write the file in blocks of eight sectors, though. Since the sectors within a block are all adjacent, a simple calculation from the block number tells you where the first sector is, and the other seven are the ones right after that one. You can still do random access to individual sectors of a file.

This trick reduces the number of addresses needed for each file's map to an eighth of what was needed before. It also has another good effect. A typical floppy disk has 2002 sectors (77 tracks of 26 sectors each), and so about 250 blocks of eight sectors. The first number requires two bytes, but the latter only requires one; this cuts the file's map size in half again. The 128-record file can now be mapped with only 16 bytes instead of 256. For just a little extra computation, the map size is now reasonable.

The second problem with mapped allocation is that directory entries in all disk systems usually have a fixed size, the same for each file. Sequential allocation directories keep a pointer and a sector count; linked directories keep one or two pointers. The file size doesn't affect the directory size. For mapped allocation you either have to deal with a complicated disk directory with entries whose sizes vary from file to file and change with time, or you have to fix a comfortable maximum size for every file and give each file that much directory space whether it needs it or not.

The solution is to do neither, or both. Fix the directory size to allow some reasonable maximum file size. The example (which is the old CP/M's standard way of doing things) is to allow up to 128 records per file. This keeps the directory entries reasonably small. But, whenever a file needs to be larger than the maximum size, just give it additional directory entries, as many as it needs.

To keep things in order, you number the individual directory entries associated with a file (e.g., FILEA1, FILEA2). In CP/M terminology each of these entries represents an extent

Bit Map:	00111111 01011100 11111100 . . . .
Sectors Free:	1,2, 9,11,15,16, 23,24, ...

Table 1. The bit map is a tight encoding of the available sectors.

Attribute	Sequential Allocation	Linked Allocation	Mapped Allocation
Random access files	Yes	No, only serial access.	Yes
Dynamic expansion of files	No, file size is fixed at creation.	Yes	Yes
Automatic reallocation of deleted files	No, Periodic compaction is required to recover available space.	Yes	Yes
Performance	Very good. Keeps all blocks from each file together. Minimizes head movement.	Not good. Related blocks tend to be scattered across disk. Much disk head movement.	Not bad. Some scattering of related blocks, but bit map tends to keep them together.

Table 2. Comparison of the three allocation methods.

of the file (see Fig. 7). This reduces the total number of files you can fit in a fixed size directory, but if you are building large files, you will have fewer of them anyway. Using these tricks, you get all the advantages of straight-mapped allocation without paying an extreme price in terms of directory size or complexity.

The only real disadvantage of this scheme is that it is harder to implement than the others. To find a record for a file, you need to calculate which extent it is in (if the file is larger than one directory entry can handle). Then you need to find the block it is in, calculate where the first sector of that block is, and then where the requested sector is relative to that one. All of these calculations require a trivial amount of computing time when compared with the disk access time so they do not significantly hurt performance, but they do complicate the operating system's code.

However, to the users of such a system, access is simple, random access is allowed, no periodic disk compactions are needed and file size can change dynamically. Digital Research's new CP/M 2.0 takes advantage of this scheme to let users allocate space at the beginning and end of a file, but not necessarily (if you don't want to) in the middle. All in all, mapped allocation of disk space is a very useful and flexible way to implement disk files.

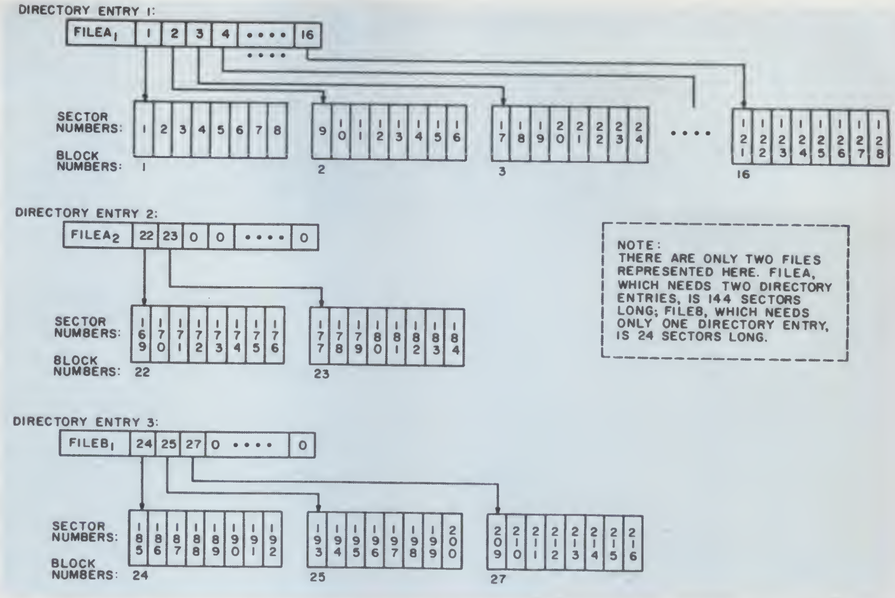


Fig. 7. Sophisticated mapped allocation scheme. Pointers are to blocks of eight sectors.

Table 2 sums up this comparison of allocation methods. It's clear that anyone trying to implement a new operating system should not use a file management system that provides anything less than the features of the mapped allocation method. ■

✓Reader Service—see page 226

Use your micro for programmed timing of multi-step processes, such as photo processing.

TRS-80 in the Darkroom

By David Busch

Program listing. Process program for the TRS-80.

```
1 'PROCESS
    David D. Busch
    515 E. Highland Avenue
    Ravenna, Ohio 44266
10 CLEAR 400
20 DIM STP$(20),AG$(20),MN$(20),SEC$(20),CHEM$(20),TEMP$(20)
30 GOSUB 1260' Pokes sound program into memory
40 GOTO 70
50 A$=INKEY$:IF A$="" GOTO 50
60 RETURN
70 CLS:PRINT:PRINT
80 PRINT "Would you like to:"
90 PRINT
100 PRINT"       1.)  Enter specs for a new process"
110 PRINT"       2.)  Run a process stored on disk"
120 PRINT
130 PRINT "          Enter choice :"
140 A$=INKEY$:IF A$="" GOTO 140
150 IF VAL(A$)<1 OR VAL(A$)>2 GOTO 140
160 ON VAL(A$) GOTO  190,460
180 '************************************
        Input Process Instructions
     ************************************
190 CLS:PRINT:PRINT
200 INPUT "How many steps are there in this process";STP
210 FOR N=1 TO STP
220 PRINT"Name of step #";N;:INPUT STP$(N)
230 INPUT "Enter chemical needed for this step";CHEM$(N)
240 PRINT "Does the ";STP$(N);" step  require agitation";:INPUT ANS$
250 IF LEFT$(AN$,1)="Y" THEN INPUT "Enter how many seconds between
 agitation";AG$(N) ELSE AG$(N)="None"
260 PRINT "How many minutes,seconds for this step.  Minutes";
:INPUT MN$(N):INPUT "Seconds";SEC$(N)
270 INPUT "Temperature required for the time given";TEMP$(N)
280 NEXT N
290 CLS:PRINT:PRINT
300 PRINT "Would you like to save this process instruction";:INPUT AN$
310 IF LEFT$(AN$,1)<>"Y" THEN RUN
320 '************************************
        Save Instructions
     ************************************
330 INPUT"Enter a file name for this process :";F$
340 OPEN "O",1,F$
350 PRINT #1,STP
360 FOR N=1 TO STP
370 PRINT #1,STP$(N);",";
380 PRINT #1,AG$(N);",";
390 PRINT #1,MN$(N);",";
400 PRINT #1,SEC$(N);",";
410 PRINT #1,CHEM$(N);",";
420 PRINT #1,TEMP$(N)
```

More

A photographic darkroom is not an ideal environment for a microcomputer—and a computer room is not the best place to develop film. But for hobbyists with multiple avocations, the two activities can co-exist if the space is kept clean, and if noxious chemicals are confined to a "wet" area away from the computer.

In several ways, a typical photographic process resembles a computer program. Certain steps, meeting a list of conditions, are carried out in a predetermined order. If other conditions arise during the "program" (e.g., the temperature starts to drop during a color process), the activity "branches" to a subroutine (adjust thermostat or raise temperature of water bath).

Process was written to permit the TRS-80 to function as a programmable timer-process instruction device. With the screen brightness turned down, and, perhaps, with a suitable 12-inch diagonal filter placed over the glass, the computer can be used even for color processes calling for total darkness or a dim safelight. The Process program provides an audible signal, thus making constant moni-

David Busch, 515 E. Highland Ave., Ravenna, OH 44266

toring of the CRT unnecessary.

The Program

This program's features include:

- Instructions for processes with up to 20 steps may be entered by the user and stored on disk.
- These steps can include directions on chemicals to be used, temperature for each step, length of time and the interval between agitation, if any.
- Both elapsed time and amount of time required for the step are displayed on the video screen.
- When an amplifier is plugged into the tape output of the TRS-80, audible signals are given at the start and end of each step, as well as the points when agitation is necessary.
- At the beginning of the process, a list of chemicals required and temperatures for each step is listed. The user is told at the end of each step which chemical should be used next.
- The user can wait between steps. If he needs to go to the bathroom while a roll of film is sitting in the stopbath, he can.

Process uses NEWDOS+, which lets you turn the TRS-80's real-time clock on and reset it to zero without exiting BASIC. TRS-DOS users can adapt the program by using whatever value of TIME$ is returned at the start of a step as a basis for comparing for elapsed time. I found it much easier to work from 00:00:00 by resetting the clock at each step. (NEWDOS 80 users should change line 780 to read: 780 TIME, 00:00:00.)

Instructions are input and saved to disk through lines 180–450. The information is stored in six arrays: STP$(N), AG$(N), MN$(N), SEC$(N), CHEM$(N) and TEMP$(N). These store data for the name of the step, agitation interval, if any, minutes, seconds, chemical used and temperature for N steps in the process. I dimensioned the arrays for 20 steps, and this can be increased by the user, but the tendency in photo technology has been to simplify processes. Color print processing has been cut from seven steps to two or three in the last 15 years.

To run a process, the user retrieves the desired instructions from the disk, using the appropriate file name that was given when the process was saved. TRI-X, E-6 or EKTA2 might be typical examples. When the file is loaded, the progam in line 590 automatically turns on the clock.

The process itself is run in lines 610–1170. A list of steps and chemi-

Listing continued.

```
430 NEXT N
440 CLOSE 1
450 GOTO 70
460 CLS:PRINT:PRINT
470 '*************************************
      Load Process from Disk
    *************************************
480 INPUT "Enter file name for desired process";F$
490 OPEN "I",1,F$
500 INPUT #1,STP
510 FOR N=1 TO STP
520 INPUT #1,STP$(N)
530 INPUT #1,AG$(N)
540 INPUT #1,MN$(N)
550 INPUT #1,SEC$(N)
560 INPUT #1,CHEM$(N)
570 INPUT #1,TEMP$(N)
580 NEXT N
590 CMD"CLOCK"
600 '***************************
       Run Process
    ***************************

610 '------- Print list of steps ------
620 CLS:PRINT:PRINT
630 PRINT " The following steps are to be followed :"
640 PRINT "Step","Chemical","Temperature"
650 PRINT
660 FOR N=1 TO STP
670 PRINT STP$(N),CHEM$(N),TEMP$(N)
680 NEXT N
690 PRINT
700 PRINT "Press any Key. Pour chemical when you get"
710 PRINT "sound or visual signal."
720 GOSUB50
730 CLS
740 '------ Run each step ------
750 FOR N=1 TO STP
760 PRINT:PRINT:PRINT"Resetting clock to zero."
770 PRINT:PRINT:PRINT "Pour chemical at signal."
780 CMD"TIME=00:00:00"
790 CLS:PRINT:PRINT
800 H4=20:GOSUB1180
810 PRINT "Pour ";CHEM$(N);" now."
820 FM=0:FS=0:FH=0:AFLAG=0
830 AG=VAL(AG$(N)):IF AG<>0 THEN AFLAG=1' Is agitation needed this step?
840 '------ Format time required this step ------
850 IF VAL(MN$(N))=0 THEN MN$(N)="00":GOTO 880
860 IF VAL(SEC$(N))=0 THEN SEC$(N)="00"
870 IF VAL(MN$(N))>0 AND VAL(MN$(N))<10 THEN MN$(N)="0"+MN$(N)
880 IF VAL(SEC$(N))>0 AND VAL(SEC$(N))<10 THEN SEC$(N)="0"+SEC$(N)
890 PRINT @ 100,"Time this step :";:PRINT @ 120,MN$(N);:PRINT @ 122,":";:PRINT @ 123,SEC$(N);
900 '------ Determine hours, minutes,seconds at finish ------
910 FS=VAL(SEC$(N))
920 IF FS>60 THEN FS=FS-60:FM=1
930 FM=FM+VAL(MN$(N))
940 IF FM>60 THEN FM=FM-60:FH=1
945 '------Test for end of allowed time ------
950 C$=RIGHT$(TIME$,8)'          C$=Current time
960 SE=VAL(RIGHT$(C$,2))'        Current seconds
980 HR=VAL(LEFT$(C$,2)):IF HR>=FH GOTO 990 ELSE GOTO 1010
990 MN=VAL(MID$(C$,4,2)):IF MN>=FM GOTO 1000 ELSE GOTO 1010
1000 IF SE>=FS GOTO 1070 ELSE GOTO 1010
1010 IF SFLAG=1 GOTO 1050
1020 '------ Agitate? -------
1030 IF AFLAG=1 THEN S1=SE/AG:S2=INT(S1):IF S1=S2 THEN SFLAG=1
1040 IF SFLAG=1 THEN PRINT @ 280," Agitate":TM=SE:H4=50:GOSUB 1180
1050 PRINT @ 280,"               ":SFLAG=0
1060 GOTO 950
1070 CLS:PRINT:PRINT
1080 H4=10:GOSUB 1180
1090 IF N=STP GOTO 1160 ELSE PRINT "The ";STP$(N);" step  is now over.  Please pour out the ";CH
EM$(N)
1100 PRINT "and prepare for the ";STP$(N+1);" step."
1110 PRINT "Have the ";CHEM$(N+1);" ready."
1120 PRINT
1130 PRINT " Press any Key when ready to continue."
1140 GOSUB 50
1150 NEXT N
1160 CLS:PRINT:PRINT:PRINT "The process is over.  Please dump the ";CHEM$(N);" ."
1170 GOTO 1170
1180 '***************************
             Sound
     ***************************
1190 POKE 32737,5
1200 FOR I=1 TO 10
1210 SO=USR0(0)
1220 FOR C=1 TO H4:NEXT C
1230 NEXT I
1240 RETURN
1250 '******************************
        Poke Sound Routine
     ******************************
1260 DEFUSR0=32736
1270 FOR N=0 TO 31
1280 READ D
1290 POKE 32736+N,D
1300 NEXT N
1310 RETURN
1320 DATA 38,100,22,1,92,14,0,62,1,0,211,255,6,16,16,254,13,40,9,29,32,246,92,47,230,3,24,237,21
,32,237,201
```

cals needed is first printed out on the CRT screen. Then, a loop of N to the number of steps (FOR N=1 to STP) begins at line 640.

Lines 840–890 format the time required for each step so that it can be displayed on the screen beneath the real-time clock's display. This routine adds zeros where appropriate, so that one minute, one second can be shown as 01:01, rather than 1:1.

At lines 950–1060 a timer subroutine is repeated until the TIME$ returned at line 950 is greater in numeric value than the time allowed for that step. Agitation at the required intervals is also activated during this portion of the program.

At the beginning of each step, and for agitation, control is sent to a subroutine at line 1180, which produces sound at the amplifier by means of a machine-language routine poked into memory when the program is first run. The signal is repeated ten times, with its sound altered by changing the value of H4, which is contained in a delay loop. The higher the value of H4, the slower the oscillation of the tone produced. This simply provides a means of letting the user differentiate between the signal used to agitate and the one that marks the end of each step.

The only bug in the program that I've been unable to overcome is the danger of spilling chemicals on the computer. Because a disk-based TRS-80 has to be the most expensive programmable darkroom timer available, I can only advise extreme caution. ■

Reader Service—see page 226

Peripheral connections are quick and easy with these modifications.

Apple Connections

By Rolf A. Deininger and Don Tujaka

Photo 1. Rear of the Apple, as modified.

Photo 2. Connection cable for paddles and joysticks.

If you move an Apple frequently, you know that disconnecting and reconnecting the peripheral equipment is a nuisance. While printers and modems are neatly handled through the 25-pin DB connectors that come with the interface and communication cards, the disk, paddles and joysticks and rf modulator output are not as cooperative. You need to insert boards in slots, thereby flexing the motherboard, and handle DIP 16-pin connectors, which are difficult to insert and whose pins become bent when frequently used.

These modifications should solve some of your problems.

Disk connection—The disk and controller board are connected with a 20-lead ribbon cable, which is cut. A 25-pin female DB connector is soldered to the cable and inserted in a slot. The male 25-pin connector, a Winchester DB25, requires no soldering.

Game I/O—On the Apple motherboard's 16-pin IC socket, only 14 pins actually carry signals; pins 9 and 16 are not connected. To bring the signals to the outside, a 16-pin DIP connector and cable is connected to a 15-pin female DB with the pin numbers corresponding; pins 9 and 16 are not connected. A removable cable then takes the signals to a small plate with a 16-pin ZIP (zero insertion pressure) DIP socket, which allows an easy exchange of paddles and joysticks. Other devices, such as a remote control box, can be directly

Rolf A. Deininger and Don Tujaka, University of Michigan, School of Public Health, Ann Arbor, MI 48109.

connected to the 15-pin connector.

TV signal output—The video output signal at location J 14 B is put through a SUP'R'MOD rf modulator, routed twice through the toroids and then brought to the outside in the form of a standard RCA phonojack. This allows straight connection to a color TV (channel 33) with a shielded cable.

Photo 1 shows the Apple from the back as modified. From right to left, the boards are:

Slot 0: Applesoft card whose toggle switch was amputated (still functional, but does not interfere with slot in back).
Slot 1: Apple clock.
Slot 2: Communications interface.
Slot 4: Serial interface to printer.
Slot 6: Disk controller.
Slot 7: Analog/digital converter, whose 40-lead ribbon cable exits through the remaining slot.

Photo 2 shows the connection cable for paddles and joysticks, and Photo 3 shows the remote control box. With these modifications Apple connections are now simple and fast.■

Photo 3. The remote control box.

René Descates * Homer * Jeremy Bentham * Plato * John Adams * Pliny the Elde

Galileo Galilei * Sigmund Freud * Michelangelo * Ludwig Van Beethoven * Leibtniz * Thomas Hobbes *

Computer\n

1: Device designed to execute a sequence of mathematical operations.

Education for the Home

Beginner's Russian

This package consists of three programs that graphically display the Cyrillic alphabet. The programs are arranged so that you progress from one to the next—building your knowledge as *you* progress. It includes instructions on proper pronunciation of the letters and even an introduction to simple Russian words.
Order No. 0136R $9.95

Everyday Russian

Everyday Russian will acquaint you with the Russian words relating to: foods, places to eat, everyday signs, and the names of common stores. You will also learn the order of the Cyrillic alphabet. Each of the three divisions of this package will teach you the words and then quiz you on comprehension. You can even practice typing in Russian, using your TRS-80 keyboard as a "Cyrillic typewriter."
Order No. 0137R $9.95

The Russian Disk

Now you can have *both* the Beginner's Russian and Everyday Russian packages on floppy disk! Requires an Expansion Interface with 16K and one disk drive.
Order No. 0212RD $24.95

Teacher

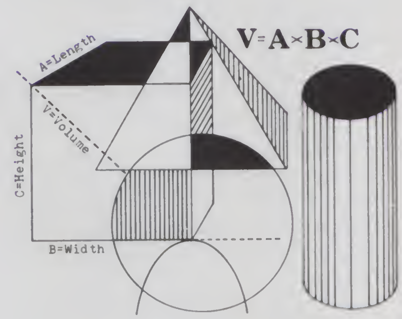
This program allows you to input any number of questions and answers. The computer will prepare tests, give quizes, provide up to three hints per question and even give (optional) graphic rewards for correct answers. Perfect for parents, teachers, or anyone faced with learning a lot of data in a short time.
Order No. 0065R $9.95

Wordwatch

Four programs for budding lexicographers, etymologists, or anyone else who uses words. In WORD RACE, you must choose the proper definitions. Find the misspelled word in HIDE N SPELL. Take a pre-recorded quiz in SPELLING BEE, in which the words are played aloud! Meet variations on proper spelling in SPELLING TUTOR.
Order No. 0111R $7.95

IQ Test

Are you smart enough to buy this package? IQ Test will administer and score an intelligence test in 30 minutes flat! There are three equivalent tests, each consisting of 35 questions, designed to test your general knowledge and problem solving abilities. Most of us claim a "touch of genius"—here's your chance to prove it!
Order No. 0157R $9.95



Archimedes' Apprentice

A tutorial software package that will teach you the formulas used to find the volume of *any* solid object. It covers parallelopipeds (cubes and rectangular solids), prisms, pyramids, cylinders, cones and spheres. It can even quiz you on how well you learned the lesson.
Order No. 0092R $9.95

Video Speed-Reading Trainer

You can increase your reading speed and comprehension. How? By practicing, that's how! This three-part program will flash characters or words on the screen, then you must echo what you saw. You can begin at a relatively slow rate, because the computer will advance your speed automatically as your speed and comprenhension increase. It will train you with numbers, letters, words and phrases.
Order No. 0100R $9.95

Typing Teacher

A complete seven-part package that guides you from familiarization with the keyboard, through typing words (and phrases), to mastery of touch typing. Your video monitor becomes a bottomless page for typing practice!
Order No. 0099R $9.95

All packages listed are for the TRS-80 Model I Level II; they require 16K of memory and are cassette-based unless otherwise indicated.

Instant Software™

PETERBOROUGH, N.H. 03458
603-924-7296

Sun Tzu Wu * Niccolo Machiavelli * Pythagoras * John Stuart Mill * Akiva *

Man your rocket ship for a journey into the fast-moving world of Apple II color graphics.

Space Race

By Robert M. Hirbernik

Try your hand at piloting a spaceship through an asteroid belt with Space Race, an exciting, fast-moving color graphics game for any Apple II computer with as little as 8K of memory.

The object is to pilot your rocket to the top of the screen as many times as you can in 90 seconds, while trying to avoid a stream of oncoming asteroids. You can play alone, or with an opponent. The program provides instructions on the screen, if desired. You need only select a single-letter response, Y or N. You also have a choice of several skill levels. The amateur level is well suited for children, or for adults playing the game for the first time. Most people will find the intermediate level comfortable, while the professional level is exceptionally challenging, even for the experienced player.

The game is played with hand controllers. Paddles zero and one control the left and right spaceships, respectively. If you rotate a paddle counterclockwise, the ship rises; if it is rotated clockwise, the ship descends. If you turn the paddle toward the middle, the ship will remain still.

```
PLOT      EQU $F800      lores plot in ROM
SETCOL    EQU $F864      lores color set
BLUE      EPZ $02        lores blue
WHITE     EPZ $0F        lores white
SPKR      EQU $C030      speaker address
          ORG $0300
          OBJ $0800
MOVE      LDA #BLUE      background color
          JSR PLOTSC     remove old asteroids
          LDX #$2F       number of asteroids (must be odd)
NEXT      DEC TABLEX,X   move left
          BPL CONT       past left side of screen?
          LDA #$27       yes, put on right side
          STA TABLEX,X   and save in table
CONT      DEX            no, get next asteroid
          INC TABLEX,X   move right
          LDA #$28       off-screen marker
          CMP TABLEX,X   past right side of screen?
          BNE CONT1      no, continue
          LDA #$00       yes, put on left side
          STA TABLEX,X   and save in table
CONT1     DEX            done with all asteroids?
          BPL NEXT       no, try again
          LDA #WHITE     yes, plot new asteroids
PLOTSC    JSR SETCOL     set to color in accumulator
          LDX #$2F       number of asteroids (must be odd)
NEXT1     LDA TABLEX,X   get X coordinate
          TAY            and save in index Y
          LDA TABLEY,X   get Y coordinate
          JSR PLOT       jump to plot routine in ROM
          DEX            done with all asteroids?
          BPL NEXT1      no, try again
          RTS            yes, back to BASIC
SOUND     LDX #$00       clear index X
SOUND1    LDA SPKR       toggle speaker
          TXA            put X into A
          TAY            transfer to Y for delay.
WAIT      DEY            done?
          BNE WAIT       no, try again
          DEX            finished with sound?
          BNE SOUND1     no, continue
          RTS            yes, return
TABLEX    EQU *          start of table (X)
TABLEY    EQU *+$30      start of table (Y)
          END
```

Listing 1. The asteroid mover routine in machine language. It was assembled using Lazer System's LISA.

Program Description

Space Race would be no fun if the asteroids didn't move simultaneously. To do this I needed speed; and Apple BASIC, fast as it is, wasn't going to cut it. So I wrote a machine-language routine to move the asteroids around, and left the main program in BASIC.

I came across two useful routines in the Apple reference manual. PLOT, located at $F800, was the easiest to find, since it was at the starting address of the monitor. Nearby was the color-setting routine, located at $F864.

For my asteroid mover, I set up buffers for the X and Y coordinates of the asteroids, and later plotted them at these coordinates. In Listing 1, you'll see that the asteroids are first erased from the screen. The ones in odd-numbered positions in the coordinate buffers are moved to the left, and those in even-numbered positions are moved to the right.

Also in Listing 1 is a tone generator, which indicates if a rocket has been wiped out by an asteroid. I put these

Robert M. Hirbernik, 46 Cornell Circle, Pueblo, CO 81005.

routines in the third page of memory so they would be out of the way of the BASIC program.

The actual game is located from line 120 to line 440 in the BASIC program (Listing 2).

In lines 120 through 180, the two ships are displayed on the screen, depending on the values of HLD1 and HLD2. These two variables are penalty timers that prevent the ships from taking off immediately after crashing. The asteroids are moved one position across the screen. If a ship and an asteroid collide, the ship will be erased from the screen and placed back at the bottom. The appropriate penalty timer (HLD1 or HLD2) will be set to 15, which is equal to a 1.5-second delay. A machine-language routine now produces the crash sound.

In lines 190 through 280, if a ship has reached the top of the screen, the score is incremented by one and the ship is put back at the bottom. The clock is now decremented by one. The scores and the time remaining are then displayed, formatted to two digits. If there is no time remaining, the ships will be placed back on the launchpad, and the players will be given time to read their scores before the game starts over.

Lines 290 through 320 read the paddles to determine the new direction of the rockets: 1 is down, 0 is stationary and −1 is up.

If a penalty for crashing is still in effect, the new direction will be zero, and the penalty timers will be decremented by one. If necessary, the ships are removed from the screen and moved to their new positions. The loop then starts over with line 120.

Fig. 1 is the flowchart for Space Race and should clear up any questions you might have about the program.

Modifications

Some interesting modifications for Space Race are possible. You might try constructing control boxes with push buttons to make the ships go up and down. More industrious types might try rewriting the entire program in machine language to make use of Apple's high-resolution graphics. ■

Listing 2. The main program written in Apple's integer BASIC.

```
100 GOSUB 20000
110 GOSUB 25000: GOSUB 10000
120 IF NOT HLD1 THEN GOSUB 370: IF NOT HLD2 THEN GOSUB 400
130 CALL 768
140 GOSUB 430
150 IF TS1 OR HLD1 THEN 170
160 GOSUB 380: Y1=37: HLD1=15: CALL 825
170 IF TS2 OR HLD2 THEN 190
180 GOSUB 410: Y2=37: HLD2=15: CALL 825
190 IF Y1 THEN 210
200 S1=S1+1: GOSUB 380: Y1=37
210 IF Y2 THEN 230
220 S2=S2+1: GOSUB 410: Y2=37
230 CLK=CLK-1
240 VTAB 21: TAB 8-(S1>9): PRINT S1;: TAB 20: IF CLK<100 THEN
    PRINT "0";: PRINT CLK/10;: TAB 33-(S2>9): PRINT S2;
250 IF CLK/10 THEN 290
260 GOSUB 380: GOSUB 410: Y1=37: Y2=37: GOSUB 370: GOSUB 400
270 FOR K=1 TO 40: CALL 768
280 FOR K1=1 TO 50: NEXT K1,K: GOTO 110
290 D1=(PDL(0)/85-1) * NOT HLD1: IF HLD1 THEN HLD1=HLD1-1
300 D2=(PDL(1)/85-1) * NOT HLD2: IF HLD2 THEN HLD2=HLD2-1
310 T1=Y1: T2=Y2: T1=T1+D1: IF T1>37 THEN T1=37: T2=T2+D2: IF T2>37
    THEN T2=37
320 IF Y1#T1 THEN GOSUB 380: IF Y2#T2 THEN GOSUB 410: Y1=T1:
    Y2=T2: GOTO 120
370 COLOR=13: GOTO 390
380 COLOR=2
390 PLOT 7,Y1: VLIN Y1+1,Y1+2 AT 6: VLIN Y1+1,Y1+2 AT 8: RETURN
400 COLOR=13: GOTO 420
410 COLOR=2
420 PLOT 32,Y2: VLIN Y2+1,Y2+2 AT 31: VLIN Y2+1,Y2+2 AT 33: RETURN
430 TS1=(SCRN(7,Y1)=13) AND (SCRN(6,Y1+1)=13) AND (SCRN(8,Y1+1)=13)
    AND (SCRN(6,Y1+2)=13) AND (SCRN(8,Y1+2)=13)
440 TS2=(SCRN(32,Y2)=13) AND (SCRN(31,Y2+1)=13) AND (SCRN(33,Y2+1)=13)
    AND (SCRN(31,Y2+2)=13) AND (SCRN(33,Y2+2)=13): RETURN
10000 TEXT: CALL -936: VTAB 7: TAB 16
10010 PRINT "SPACE RACE": TAB 16: PRINT "----- ----"
10020 VTAB 18: TAB 12: PRINT "INSTRUCTIONS? (Y/N)"
10030 POKE -16368,0
10040 IF PEEK(-16384)=206 THEN 10060
10050 IF PEEK(-16384)=217 THEN 10220: GOTO 10040
10060 CALL -936: VTAB 3: TAB 7
10070 PRINT "WHAT IS YOUR SKILL LEVEL?": VTAB 8: TAB 12
10080 PRINT "A)  AMATEUR": VTAB 12: TAB 12
10090 PRINT "B)  INTERMEDIATE": VTAB 16: TAB 12
10100 PRINT "C) PROFESSIONAL": POKE -16368,0: SK=0
```

More

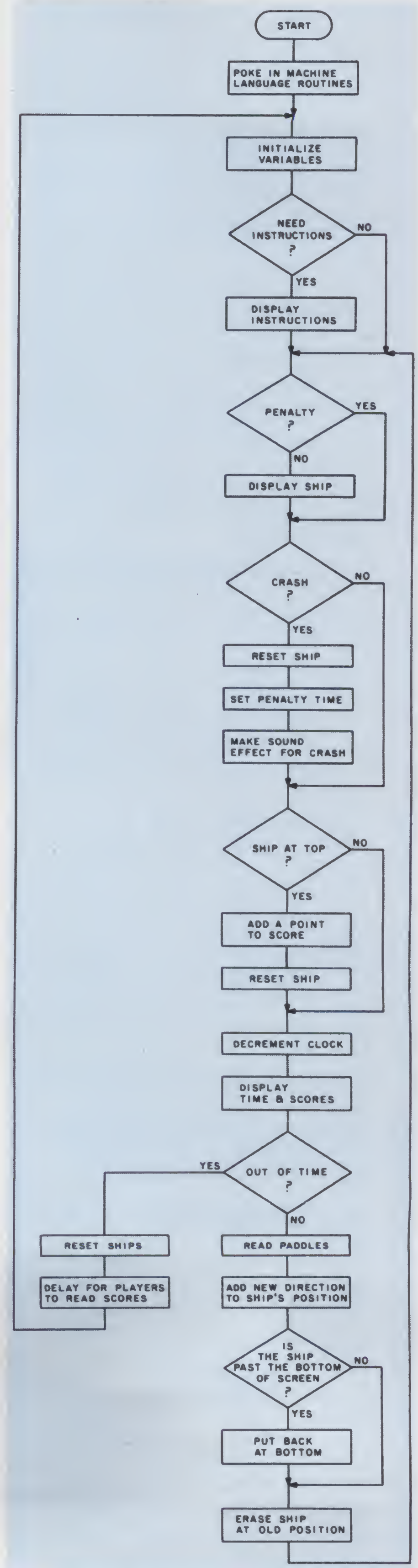


Fig. 1. Flowchart for Space Race.

Listing 1 continued.

```
10110 IF PEEK(-16384)=193 THEN SK=15
10120 IF PEEK(-16384)=194 THEN SK=31
10130 IF PEEK(-16384)=195 THEN SK=47
10140 IF SK=0 THEN 10110: POKE 774,SK: POKE 810,SK
10150 FOR M=0 TO SK: POKE M+839, RND(40)
10160 POKE M+887,RND(19)*2: NEXT M
10170 GR: COLOR=2: FOR M=0 TO 39
10180 VLIN 0,39 AT M: NEXT M
10190 VTAB 21: TAB 7: PRINT "0";: TAB 32: PRINT "0";
10200 VTAB 23: TAB 16: POKE 50,63: PRINT "SPACE RACE"
10210 TAB 16: PRINT "----- ----";; POKE 50,255: RETURN
10220 CALL -936: VTAB 2: TAB 16: POKE 50,63: PRINT "SPACE RACE":
      POKE 50,255
10230 VTAB 4: PRINT "THE OBJECT OF THIS GAME IS TO PILOT YOUR": PRINT
      "SHIP THROUGH AN ASTEROID BELT MORE TIMES"
10240 PRINT "THAN YOUR OPPONENT WITHIN 90 SECONDS.   ": PRINT
10250 PRINT "THE SHIPS ARE CONTROLLED BY USE OF THE  ": PRINT "PADDLES.
      THE CENTER POSITION WILL KEEP  "
10260 PRINT "THE SHIP STILL. ROTATING THE PADDLE LEFT": PRINT "OR
      RIGHT WILL MOVE THE SHIP UP AND DOWN."
10270 VTAB 23: PRINT "PRESS 'RETURN' WHEN READY.": POKE -16368,0
10280 IF PEEK(-16384)<128 THEN 10280: GOTO 10060
20000 REM  STAR MOVER
20010 POKE 768,169: POKE 769,2: POKE 770,32: POKE 771,38
20020 POKE 772,3: POKE 773,162: POKE 774,47: POKE 775,222
20030 POKE 776,71: POKE 777,3: POKE 778,16: POKE 779,5
20040 POKE 780,169: POKE 781,39: POKE 782,157: POKE 783,71
20050 POKE 784,3: POKE 785,202: POKE 786,254: POKE 787,71
20060 POKE 788,3: POKE 789,169: POKE 790,40: POKE 791,221
20070 POKE 792,71: POKE 793,3: POKE 794,208: POKE 795,5
20080 POKE 796,169: POKE 797,0: POKE 798,157: POKE 799,71
20090 POKE 800,3: POKE 801,202: POKE 802,16: POKE 803,227
20100 POKE 804,169: POKE 805,15: POKE 806,32: POKE 807,100
20110 POKE 808,248: POKE 809,162: POKE 810,47: POKE 811,189
20120 POKE 812,71: POKE 813,3: POKE 814,168: POKE 815,189
20130 POKE 816,119: POKE 817,3: POKE 818,32: POKE 819,0
20140 POKE 820,248: POKE 821,202: POKE 822,16: POKE 823,243
20150 POKE 824,96
20160 REM  SOUND FOR CRASH
20170 POKE 825,162: POKE 826,0: POKE 827,173: POKE 828,48
20180 POKE 829,192: POKE 830,138: POKE 831,168: POKE 832,136
20190 POKE 833,208: POKE 834,253: POKE 835,202: POKE 836,208
20200 POKE 837,245: POKE 838,96
20210 RETURN
25000 Y1=Y2=HLD1=HLD2=CLK=S1=S2=T1=T2
25010 Y1=37: Y2=37: HLD1=0: HLD2=0: CLK=900: S1=0: S2=0
25020 RETURN
30000 REM
30010 REM *********************
30020 REM *    SPACE RACE     *
30030 REM *********************
30040 REM *                   *
30050 REM * R. M.  HIRBERNIK  *
30060 REM *                   *
30070 REM *  MARCH 27, 1980   *
30080 REM *                   *
30090 REM *********************
32767 END
```

Reader Service—see page 226

Faced with a desire for additional capability from BASIC, the author took a closer look at H8 user subroutines and discovered how simple the interface between them and BASIC can be.

Enhancing H8 BASIC

By Edgar C. Howell

Are you tired of being upstaged by the new kid on the block? Do you still believe in your trusty old H8, even as you wish for tape I/O or exotic functions not reasonably implemented in BASIC?

If you initially reacted to the limitations of Heath's BASIC as I did, don't wait as long to discover how easily they can be overcome.

After reading that only one argument can be passed to a user subroutine, I gave up on the idea of writing subroutines. And the lack of tape record I/O instructions precluded routines to manipulate files, such as balancing my checkbook. So, having bought and built the hardware to play with the assembler anyhow, I mentally relegated this BASIC to the realm of games.

Then one day came the need for terminal input, saving the file, sorting and totals. This was too grim to consider exclusively in assembler, so I took a closer look at BASIC.

I found that although the H8 BASIC user subroutine call formally restricts you to a single argument and a single return value, these can be memory addresses, and you can peek and poke additional values as well.

If you are using Extended Benton Harbor BASIC version 10.01.01 on an H8/H9 with 16K, these routines will enable you to read files created by the text editor. You could use additional panel monitor calls to write them as well, but once you see how easy the interface is, you might consider doing something really useful.

Be careful if your environment is different. Note *what* is done; *how* it is done may not be applicable to your system. For example, if you have 32K and follow the directions below, you will waste half your memory. Step 2 of Table 1 overrides the configured high memory to reserve for user subroutines the last 1K of a 16K system—or 17K of a 32K system!

Loading BASIC and a Subroutine

Table 1 shows one way to combine BASIC and a user subroutine in memory. In version 10.01.01 a cold start jumps to 041.156 (offset octal as usual) to initialize the stack pointer. The value to be changed (137.377) is the operand of a three-byte instruction, the high memory address entered when you initially configured BASIC.

In this case, subtracting 4 from the high-order byte reduces the initial value of the stack and reserves 1024 bytes of memory above it. Since loading changes the contents of the program counter, step 4 merely ensures that it points to the entry point of BASIC.

While this procedure proved adequate in testing the user subroutine discussed below, as always it would be a better idea to follow Heath's directions to configure a copy of BASIC once you know how much memory to reserve for your subroutine. If you do this, step 2 becomes unnecessary. As step 5 implies, this procedure should be done before executing BASIC; if done afterwards, the FREE function will not correctly reflect the new high memory limit. That may have any number of other consequences.

The BASIC Program

Listing 1 is a BASIC program that prints TED-8 records given to it by a user subroutine. Lines 100–110 tell BASIC where to begin executing the subroutine when called in line 210. This address could have been set as part of the procedure in Table 1, but dynamically poking the address allows you to select one of several user subroutines.

If you have two tape drives, you might want two subroutines—one for input, one for output. Then each time you want to swap from input to output or vice versa, you would only need to poke the corresponding subroutine address before calling USR.

Lines 120–130 ensure that the subroutine always reads a block from tape when you run the program. Otherwise, if the program is interrupted by CTRL-C, the subroutine will not necessarily read but will continue with the next line in the input buffer when you run again.

```
100 POKE 17267,0          :REM GIVE BASIC ADDRESS
110 POKE 17268,92         :REM    OF USER SUBROUTINE
120 POKE 23628,0          :REM ZERO ADDR LAST BYTE READ TO
130 POKE 23629,0          :REM    FORCE INITIAL READ IF RE-RUN
200 FOR I = 1 TO 1000     :REM LOTS OF TIMES
210 K = USR(0)            :REM ADDRESS NEXT LINE FROM TAPE
220 L = PEEK(K)           :REM NEXT CHARACTER THIS LINE
230 K = K + 1             :REM BUMP CHARACTER POINTER
240 IF L < 128 THEN 260   :REM JUMP IF NOT COMPRESSED BLANKS
250 FOR M = 129 TO L:PRINT " ";:NEXT M:GOTO 220
260 IF L > 0 THEN PRINT CHR$(L);:GOTO 220
270 PRINT                 :REM END OF LINE
280 NEXT I
999 END
```

Listing 1. Printing text input by user subroutine.

Edgar C. Howell, 2909 Tapo St., Simi, CA 93063

1. Load BASIC
2. Change the address of 041.157 to the desired high-memory address; e.g., to reserve 1K in a 16K system

change	041.157	377
	041.160	137
to	041.157	377
	041.160	133

3. Load assembled user subroutine(s)
4. Set PC to 040.100
5. Depress GO

Table 1. Appending user subroutines to H-8 BASIC.

This also demonstrates a simple way around the restriction to a single argument or return value, since the zero poked is essentially an additional argument telling the subroutine to ignore the current contents of its input buffer and read another block from tape. POKE can be used to perform initialization prior to the first call as well as to pass multiple arguments prior to each call, and PEEK will retrieve multiple return values.

In line 210 a user subroutine or USR call obtains the address of a line of text in the input buffer. Note that this is an address and not a character string. The loop from 220 to 260 prints the line, expanding compressed blanks, until a null is found signifying the end of the line. A similar loop could have been used to construct a character string by concatenating instead of printing.

The Assembler Subroutine

The assembler routine in Listing 2 communicates with the tape drive via PAM-8 subroutines, which are quite well documented, as usual, and available at no extra charge in ROM. The routine is ORGed to the first memory location above the stack, the same address poked in Listing 1, lines 100–110.

Use of Memory

The routine uses otherwise unused memory to hold a couple of addresses and store the data as it is input from tape. MADDR is a label for the address of the first available byte. The first two bytes there hold the address of the memory location in which the last byte read from tape is stored.

Since each line of text is terminated by a null, after the first line the routine merely looks for successive nulls and gives BASIC the address of the following byte—unless it would go beyond this last byte read, in which

```
* ADDRESSES OF PAM-8 ROUTINES
*
SRS     EQU  002265A     SCAN RECORD START
RNB     EQU  002331A     READ NEXT BYTE
TFT     EQU  002133A     TURN OFF TAPE
*
        ORG  23552
        PUSH PSW
        PUSH H
        PUSH D
        PUSH B
*
        LHLD MADDR+2     ADDR LAST LINE RETURNED TO BASIC
LOOP1   EQU  *
        INX  H           LOOK
        MOV  A,M           FOR
        ORA  A               END OF
        JNZ  LOOP1             LINE
*
        INX  H           ADDR 1ST BYTE NEXT LINE
        LDA  MADDR       GIVE BASIC
        SUB  L             THIS ADDRESS
        LDA  MADDR+1         IF NOT BEYOND
        SBB  H                 LAST BYTE IN
        JNC  XINPT               INPUT BUFFER
*
        CALL SRS         SCAN RECORD START
        MOV  B,H         DATA
        MOV  C,A           COUNT
*
        LXI  H,MADDR+4-1  ADDR INPUT BUFFER - 1
*
NXTIN   EQU  *
        CALL RNB         READ NEXT BYTE
        INX  H           INTO
        MOV  M,A           BUFFER
*
        DCX  B           DECREMENT
        MOV  A,B           AND TEST
        ORA  C               DATA COUNT
        JNZ  NXTIN
*
        CALL TFT         TURN OFF TAPE
*
        SHLD MADDR       KEEP ADDR LAST BYTE READ
        INX  H           ENSURE TEST FOR
        MVI  M,0           END OF LINE WORKS
*
        LXI  H,MADDR+4   ADDR 1ST LINE IN BUFFER
XINPT   EQU  *
        SHLD MADDR+2     KEEP ADDR LINE BEING RETURNED
*
        POP  B           ADDR
        MOV  D,B           RETURN
        MOV  E,C             PARAMETER
*
        XCHG             ADDRESSABLE BY M
        MVI  M,0         GIVE
        INX  H             BASIC
        MOV  M,E             ADDR
        INX  H                 NEXT
        MOV  M,D                 LINE
        INX  H                     IN
        MVI  M,217A                  BUFFER
*
        POP  D
        POP  H
        POP  PSW
*
        RET
*
MADDR   EQU  *           AVAILABLE MEMORY
        DW   0           ADDR LAST BYTE READ
        DW   0           ADDR LAST LINE RETURNED TO BASIC
        END  0
```

Listing 2. User subroutine to input text for BASIC.

```
100 GOSUB 1000             :REM INITIALIZE NUMERIC KEYPAD
200 FOR I = 1 TO 10        :REM GET AND PRINT 10 "DIGITS"
210 J = USR(0)             :REM NEXT KEYSTROKE INTO MEMORY
220 K = PEEK(8214)         :REM FETCH IT
230 PRINT K;               :REM SHOW AND TELL TIME
240 NEXT I
250 STOP
1000 POKE 17267,0          :REM GIVE BASIC ADDRESS
1010 POKE 17268,92         :REM   OF USER SUBROUTINE
1100 FOR I = 23552 TO 23558
1110 READ J                :REM POKE MACHINE LANGUAGE ROUTINE
1120 POKE I,J              :REM   315 260 003    CALL  RCK
1130 NEXT I                :REM   062 026 040    STA   RCKA
1140 RETURN                :REM   311            RET
1200 DATA 205,176,3,50,22,32,201
```

Listing 3. Poking a user subroutine to input from the front panel keypad.

case another block of data is read.

The next two bytes (at MADDR+2) hold the address of the line previously passed to BASIC. Scanning for the next null always begins at the location whose address is stored here. And the remainder of the available memory (at MADDR+4) is used as an input buffer.

By the way, the routine is meant only to demonstrate user subroutines and is not complete, since it omits such functions as error processing, test for end of file and return codes. It can't even tell if it has a TED-8 file, and will happily read BASIC text.

Parameter Passing

Parameter passing is a bit sticky if your mind, like mine, doesn't function too well in floating-point binary. Upon passing control to the subroutine, BASIC leaves in register pair BC the address of a four-byte area used to pass a value to the subroutine or back to BASIC. Suffice it to say that 217_8 is an exponent adequate to tell BASIC that the value returned in the other three bytes contains 16 bits (including a sign-bit) to the left of the implicit binary point, i.e., can hold a high-memory address less than 32K. For details see p. 5–101 in the *Software Reference Manual.*

A Numeric Keypad for the H-8?

In conclusion, for anyone still not convinced that user subroutines can be easy, I submit in Listing 3 a routine with which you can quickly check it out and which you should be able to decipher—with a little help from our friends at Heath. While it will never replace the PAD function, execution of the subroutine at 1000 prepares BASIC to input from the front panel's numeric keypad. But watch out for that automatic repeat!

So there you have it—not at all as difficult as you might have anticipated. Just make BASIC reserve a little high memory for you, *assemble the* user subroutine at or above that address, load the assembled routine along with BASIC and make sure BASIC knows where to begin executing it.■

NEW! TPM* for TRS-80 Model II
NEW! System/6 Package
Computer Design Labs

Z80* Disk Software

We have acquired the rights to all TDL software (& hardware). TDL software has long had the reputation of being the best in the industry. Computer Design Labs will continue to maintain, evolve and add to this superior line of quality software.

— Carl Galletti and Roger Amidon, owners.

Software with Manual/Manual Alone

All of the software below is available on any of the following media for operation with a Z80 CPU using the CP/M* or similar type disk operating system (such as our own TPM*).

for TRS-80* CP/M (Model I or II)
for 8" CP/M (soft sectored single density)
for 5¼" CP/M (soft sectored single density)
for 5¼" North Star CP/M (single density)
for 5¼" North Star CP/M (double density)

BASIC I

A powerful and fast Z80 Basic interpreter with EDIT, RENUMBER, TRACE, PRINT USING, assembly language subroutine CALL, LOADGO for "chaining", COPY to move text, EXCHANGE, KILL, LINE INPUT, error intercept, sequential file handling in both ASCII and binary formats, and much, much more. It runs in a little over 12 K. An excellent choice for games since the precision was limited to 7 digits in order to make it one of the fastest around. $49.95/$15.

BASIC II

Basic I but with 12 digit precision to make its power available to the business world with only a slight sacrifice in speed. Still runs faster than most other Basics (even those with much less precision). $99.95/$15.

BUSINESS BASIC

The most powerful Basic for business applications. It adds to Basic II with random or sequential disk files in either fixed or variable record lengths, simultaneous access to multiple disk files, PRIVACY command to prohibit user access to source code, global editing, added math functions, and disk file maintenance capability without leaving Basic (list, rename, or delete). $179.95/$25.

ZEDIT

A character oriented text editor with 26 commands and "macro" capability for stringing multiple commands together. Included are a complete array of character move, add, delete, and display function. $49.95./$15.

ZTEL

Z80 Text Editing Language - Not just a text editor. Actually a language which allows you to edit text and also write, save, and recall programs which manipulate text. Commands include conditional branching, subroutine calls, iteration, block move, expression evaluation, and much more. Contains 36 value registers and 10 text registers. Be creative! Manipulate text with commands you write using Ztel. $79.95/$25.

TOP

A Z80 Text Output Processor which will do text formatting for manuals, documents, and other word processing jobs. Works with any text editor. Does justification, page numbering and headings, spacing, centering, and much more! $79.95/$25.

MACRO I

A macro assembler which will generate relocateable or absolute code for the 8080 or Z80 using standard Intel mnemonics plus TDL/Z80 extensions. Functions include 14 conditionals, 16 listing controls, 54 pseudo-ops, 11 arithmetic/logical operations, local and global symbols, chaining files, linking capability with optional linker, and recursive/reiterative macros. This assembler is so powerful you'll think it is doing all the work for you. It actually makes assembly language programming much less of an effort and more creative. $79.95/$20.

MACRO II

Expands upon Macro I's linking capability (which is useful but somewhat limited) thereby being able to take full advantage of the optional Linker. Also a time and date function has been added and the listing capability improved. $99.95/$25.

LINKER

How many times have you written the same subroutine in each new program? Top notch professional programmers compile a library of these subroutines and use a Linker to tie them together at assembly time. Development time is thus drastically reduced and becomes comparable to writing in a high level language but with all the speed of assembly language. So, get the new CDL Linker and start writing programs in a fraction of the time it took before. Linker is compatible with Macro I & II as well as TDL/Xitan assemblers version 2.0 or later. $79.95/$20.

DEBUG I

Many programmers give up on writing in assembly language even though they know their programs would be faster and more powerful. To them assembly language seems difficult to understand and follow, as well as being a nightmare to debug. Well, not with proper tools like Debug I. With Debug I you can easily follow the flow of any Z80 or 8080 program. Trace the program one step at a time or 10 steps or whatever you like. At each step you will be able to see the instruction executed and what it did. If desired, modifications can then be made before continuing. It's all under your control. You can even skip displaying a subroutine call and up to seven breakpoints can be set during execution. Use of Debug I can pay for itself many times over by saving you valuable debugging time. $79.95/$20.

DEBUG II

This is an expanded debugger which has all of the features of Debug I plus many more. You can "trap" (i.e. trace a program until a set of register, flag, and/or memory conditions occur). Also, instructions may be entered and executed immediately. This makes it easy to learn new instructions by examining registers/memory before and after. And a RADIX function allows changing between ASCII, binary, decimal, hex, octal, signed decimal, or split octal. All these features and more add up to give you a very powerful development tool. Both Debug I and II must run on a Z80 but will debug both Z80 and 8080 code. $99.95/$20.

ZAPPLE

A Z80 executive and debug monitor. Capable of search, ASCII put and display, read and write to I/O ports, hex math, breakpoint, execute, move, fill, display, read and write in Intel or binary format tape, and more! on disk

APPLE

8080 version of Zapple

NEW! TPM now available for TRS-80 Model II!

TPM*

A NEW Z80 disk operation system! This is not CP/M*. It's better! You can still run any program which runs with CP/M* but unlike CP/M* this operating system was written specifically for the Z80* and takes full advantage of its extra powerful instruction set. In other words its not warmed over 8080 code! Available for TRS-80* (Model I or II). Tarbell, Xitan DDDC, SD Sales "VERSA-FLOPPY", North Star (SD&DD), and Digital (Micro) Systems. $79.95/$25.

SYSTEM MONITOR BOARD (SMB II)

A complete I/O board for S-100 systems. 2 serial ports, 2 parallel ports, 1200/2400 baud cassette tape interface, sockets for 2K of RAM, 3-2708/2716 EPROM's or ROM, jump on reset circuitry. Bare board $49.95/$20.

ROM FOR SMB II

2KX8 masked ROM of Zapple monitor. Includes source listing $34.95/$15.

PAYROLL (source code only)

The Osborne package. Requires C Basic 2.
5" disks $124.95 (manual not included)
8" disks $ 99.95 (manual not included)
Manual $20.00

ACCOUNTS PAYABLE/RECEIVABLE (source code only)

By Osborne, Requires C Basic 2
5" disks $124.95 (manual not included)
8" $99.95 (manual not included)
Manual $20.00

GENERAL LEDGER (source code only)

By Osborne. Requires C Basic 2
5" disks $99.95 (manual not included)
8" disks $99.95 (manual not included)
Manual $20.00

C BASIC 2

Required for Osborne software. $99.95/$20.

SYSTEM/6

TPM with utilities, Basic I interpreter, Basic E compiler, Macro I assembler, Debug I debugger, and ZEDIT text editor.
Above purchased separately costs $339.75
Special introductory offer: Only $179.75 with coupon!!

$160.

This Coupon Is Worth One Hundred And Sixty Dollars Toward The Full Price Of The SYSTEM/6 Package System/6 with this coupon is only $179.95. This is a limited time offer.

$160.00

ORDERING INFORMATION

Visa, Master Charge and C.O.D. O.K. To order call or write with the following information.

1. Name of Product (e.g. Macro I)
2. Media (e.g. 8" CP/M)
3. Price and method of payment (e.g. C.O.D.) include credit card info. if applicable.
4. Name, Address and Phone number.
5. For TPM orders only: Indicate if for TRS 80, Tarbell, Xitan DDDC, SD Sales (5¼" or 8"). ICOM (5¼" or 8"), North Star (single or double density) or Digital (Micro) Systems.
6. N.J. residents add 5% sales tax.

Manual cost applicable against price of subsequent software purchase in any item except for the Osborne software.

For information and tech queries call
609-599-2146

For phone orders ONLY call toll free
1-800-327-9191
Ext. 676
(Except Florida)

OEMS

Many CDL products are available for licensing to OEMs. Write to Carl Galletti with your requirements.

* Z80 is a trademark of Zilog
* TRS-80 is a trademark for Radio Shack
* TPM is a trademark of Computer Design Labs. It is not CP/M*
* CP/M is a trademark of Digital Research

Prices and specifications subject to change without notice.

DEALER INQUIRIES INVITED.

COMPUTER DESIGN LABS

18

342 Columbus Avenue
Trenton, N.J. 08629


SAVE ON
TRS-80's
from
Pan American
Electronics, Inc.
Model I
A Radio Shack
AUTHORIZED·SALES CENTER
We Have
DISCOUNTS
FREE SHIPPING
and a
TOLL FREE ORDER NUMBER
800/531-7466
Texas and Principle Number 512/581-2765
Telex Number 767339
Department K
1117 Conway
Mission, Texas 78572
MasterCard
VISA®
Model II
Model III
Color Computer
Pocket Computer
✓71

This barebones system provides instant access to information, and a way to keep your books balanced.

A Minimum Accounting System

By Gene Embry

The 18th century German poet Johann Wolfgang von Goethe wrote that double entry accounting "is one of the most beautiful inventions of the human spirit, and every good businessman should use it in his economic undertakings."

While today's authors of business software might not describe it in such poetic terms, double entry is still used to help modern merchants keep their books.

This article will explain how to use a minimum accounting system to set up a chart of accounts (COA) and a method of keeping the books in balance.

The Rules

The accounting rules this program follows are simple: The sum of all assets must equal the sum of all liabilities plus the sum of all equities, and each transaction must affect at least two accounts.

Assume we decided to form a company called SuperSoftware, Inc., to develop business software. Our first job is to develop an accounting system. We study our present needs and future requirements, and lay down the following guidelines for our system:

- It must be simple to use.
- All transactions will instantly update the COA.
- It must be easily expandable.
- It should provide an audit trail.
- It should provide for end-of-period procedures.

The second item means that we can check our profit and loss (P&L) statement and/or balance sheet after each transaction is entered into the system, and the books will be balanced. The fourth item lets us check on human input errors.

After some serious thought, we arrive at the major division in Table 1 for our (COA). A list of the individual accounts is shown in Table 2.

Our System

The system we decided to purchase comprises

- A 6800-based CPU with 32K RAM.
- A video terminal.
- A printer.
- A dual eight-inch floppy disk.

Smoke Signal Broadcasting Co. disk operating system (version 51) and Computerware's BASIC (version 8.5) fill our immediate needs for a good DOS and a flexible high-level language.

First, we must determine what information we need to save. We'll need a general data file to store variables that will be passed between programs, and other variables such as the number of records used in a data file, dates, identification numbers and flags. A file of 25 records, with each record containing two numerics and one string variable will be sufficient for our needs. This file will be named MASTER.DAT.

Account No.	Type of Account
100–199	Asset
200–299	Liability
300–399	Equity
400–499	Sales
500–599	Cost of merchandise sold
600–699	Expenses
700–799	Other income
800–899	Other expenses

Table 1. Major divisions of COA accounts.

Each account of the COA will be stored in a file named CHART.DAT. Each record will contain four numerics and one string.

One file will be needed to store transactions. This file, TRANS.DAT, will contain 50 records, and each record will have five numerics and one string.

The assignment of the locations used in the file MASTER.DAT is shown in Table 3. Note that record 2, field 1, shows a formatted date stored as a numeric, which is configured as eight digits with the first four digits the year, the next two digits the month and the last two the day of the month. This method is always used and has many advantages.

Computerware's technique of creating random files is in accordance with the following formula:

bytes per record $= 6*N + S*(L + 1)$.

N = number of numerics per record,
S = number of strings per record, and
L = length of each string.

We will format each record in the file CHART.DAT in the following manner:

A = Account number
A$ = Account description
B = Current balance
C = Year-to-date balance
D = Spare

Gene Embry, Route 1, Box 151-H, Morrisville, NC 27560.

Table 2. Chart of accounts.

ACCT.	#	DESCRIPTION	CURRENT	BALANCE
1	100	CHECKING		95094.05
2	103	PETTY CASH		0.00
3	106	PREPAID INSURANCE		0.00
4	107	PREPAID INTEREST		0.00
5	108	INSURANCE - CASH VALUE		0.00
6	110	A/R - TRADE		75.00
7	111	A/R - EMPLOYEES		0.00
8	120	INVENTORY		0.00
9	150	LAND		0.00
10	160	DEPRECIABLE ASSETS		5000.00
11	161	LESS DEPRECIATION CLAIMED		0.00
12	180	DEFERRED INTEREST EXPENSE		0.00
13	185	ORGANIZATIONAL EXPENSE		0.00
14	200	ACCOUNTS PAYABLE		0.00
15	205	FICA WITHHELD		0.00
16	206	FED. TAX WITHHELD		0.00
17	207	STATE TAX WITHHELD		0.00
18	208	STATE - UNEMP TAX		0.00
19	209	FED. UNEMP TAX		0.00
20	217	ACCRUED INTEREST PAYABLE		0.00
21	220	NOTE #1 PAYABLE		0.00
22	222	NOTE #2 PAYABLE		0.00
23	300	CAPITAL		100000.00
24	310	RETAINED EARNINGS		169.05
25	399	NET PROFIT		0.00
26	400	CASH - SALES	0.00	100.00
27	401	SALES - ON ACCOUNT	0.00	75.00
28	500	COST OF MDSE. SOLD	0.00	0.00
29	600	ADVERTISING & SALES PROM.	0.00	0.00
30	602	ACCOUNTING & AUDIT FEES	0.00	0.00
31	604	BANK SERVICE CHARGE	0.00	0.00
32	606	DATA PROCESSING	0.00	0.00
33	610	DEPRECIATION	0.00	0.00
34	614	DONATION	0.00	0.00
35	616	DUES & SUBSCRIPTION	0.00	0.00
36	617	FREIGHT	0.00	5.95
37	620	INSURANCE	0.00	0.00
38	624	INTEREST	0.00	0.00
39	630	LEGAL	0.00	0.00
40	634	MAINTENANCE	0.00	0.00
41	637	INVENTORY SHORTAGE	0.00	0.00
42	638	OFFICE SUPPLIES	0.00	0.00
43	640	PAYROLL	0.00	0.00
44	648	PAYROLL TAXES	0.00	0.00
45	658	POSTAGE	0.00	0.00
46	660	RENT	0.00	0.00
47	666	TELEPHONE & TELEGRAPH	0.00	0.00
48	668	TRAVEL & ENTERTAINMENT	0.00	0.00
49	676	UTILITIES	0.00	0.00
50	700	INTEREST INCOME	0.00	0.00
51	710	RENTAL INCOME	0.00	0.00
52	720	CASH OVER OR SHORT	0.00	0.00
53	730	SALE OF ASSETS	0.00	0.00
54	805	BAD DEBTS	0.00	0.00
55	810	PENALITIES	0.00	0.00
56	820	COST OF ASSETS SOLD	0.00	0.00
57	830	MISCELLANEOUS EXPENSES	0.00	0.00

MASTER.DAT File Contents
March 1, 1980

REC.#		Description & Value	
1	RESERVED=0	RESERVED=0	MONTH=January
2	TODAY'S DATE FORMATTED=19800301	JULIAN DAY=2444301	MONTH=February
3	RESERVED=0	RESERVED=0	MONTH=March
4	RESERVED=0	RESERVED=0	MONTH=April
5	RESERVED=0	RESERVED=0	MONTH=May
6	RESERVED=0	# IN TRANS.DAT=6	MONTH=June
7	PROFIT - P&L=0	END OF MONTH FLAG=0	MONTH=July
8	RESERVED=0	RESERVED=0	MONTH=August
9	RESERVED=0	RESERVED=0	MONTH=September
10	RESERVED=0	SPARE=0	MONTH=October
11	RESERVED=0	RESERVED=0	MONTH=November
12	RESERVED=0	RESERVED=0	MONTH=December
13	SPARE=0	SPARE=0	NAME=SuperSoftware, Inc.
14	SPARE=0	SPARE=0	TODAY-READABLE=March 1, 1980
15	SPARE=0	SPARE=0	SPARE=
16	SPARE=0	SPARE=0	SPARE=
17	SPARE=0	SPARE=0	SPARE=
18	SPARE=0	SPARE=0	SPARE=
19	SPARE=0	SPARE=0	SPARE=
20	SPARE=0	SPARE=0	SPARE=
21	SPARE=0	SPARE=0	RESERVED=
22	SPARE=0	SPARE=0	ADDRESS=312 Green Street
23	SPARE=0	SPARE=0	CITY-STATE-ZIP=Carpenter, NC 27560
24	SPARE=0	SPARE=0	RESERVED=
25	SPARE=0	SPARE=0	RESERVED=

Table 3. MASTER.DAT data file.

Program Name	Purpose
BEGIN	Operator input of today's date.
LEDGER.BAS	The main menu program.
INCOME.BAS	Prints the income statement (P&L).
BALANC.SHT	Prints the balance sheet.
TRANS.BAS	Operator input of all transactions.
TRANS.PRT	Prints the transaction report.

Table 4. Main programs.

Each record in file TRANS.DAT will be stored as follows:

A = The account number to take money from
B = The account number to transfer the money to
A$ = Description of the transaction
C = Date of transaction
D = Dollar amount of the transaction
E = The check number, if the transaction is a check.

Next we'll determine what programs we'll need and how the programs will interact with each other. A menu-driven system will be best. The main working programs are shown in Table 4.

Four other programs will be needed to complete our minimum accounting package. One program, INIT.BAS, will be used only one time to create the three data files and to get the name and address of our company. All the account numbers and descriptions are contained in DATA statements. You may add or delete accounts freely. The three constraints are that the account numbers be stored in ascending order, that no commas be used in the description, and that accounts be limited to 255. After INIT.BAS is finished, program BEGIN is called and executed.

Another program, CHART.COR, will let you insert commas to your heart's content. CHART.COR will also let you obtain a listing of all the information contained in the file CHART.DAT and to make corrections. CHART.COR will only be called if there is an error in the books. Following the completion of CHART.COR, it will return to the main menu program, LEDGER.BAS.

RESET.COA is a short program to set all money fields in CHART.DAT to zero. It is never called by any of the programs, and is used mainly during

the testing of the system software.

The final program, MASTER.CHG, is used to get a printout of the data in MASTER.DAT and to change values in the file. It may be called using LEDGER.BAS, and returns to LEDGER.BAS when completed.

Key Point

How do we determine if the books are balanced? The main purpose of an income statement is to report the current period profit or loss. The net profit, as determined from the income statement, is equal to the sum of all sales plus other income minus the cost of merchandise sold minus operating expenses minus other expenses. Note that this will include only accounts that are numbered 400 and greater.

The balance sheet is a historical money summary of everything that has happened to the company since its beginning. We will define the retained earnings account (310) as the sum of all end-of-period profits and losses for the corporation. The net profit (399), as determined from the balance sheet, will be the sum of all accounts from 100 to 199 minus the sum of all accounts from 200 to 299 minus the sum of all accounts from 300 to 398. After calculating net profits, using two completely different sets of numbers, we compare them. If they are equal, the books are in balance.

The BEGIN Program

BEGIN should be the first program called each day. It gets today's date from the operator and calculates the Julian day based on this input. You will find the Julian day calculation a very nice way to age accounts receivable and accounts payable and to calculate interest. Various tests on the inputs are made, but they are by no means complete. The BASIC statement DLM = OFF allows commas inside a string. In all programs, we will be using the default value of STRING = 32. This program is shown in Listing 1.

The LEDGER.BAS Program

This is the main menu program. See Sample 1. Note the generalized technique of printing out the menu in lines 150–156 (see Listing 2). The addition or deletion of items to the menu only requires insertion or deletion of a DATA line. The colon (:) is the equivalent of a remark statement. Lines 1030 to 1050 show some future

Sample 1. Main menu.

```
                         SuperSoftware, Inc.
                           March 1, 1980

                      Minimum Accounting System

    1 . Post Transactions                     2 . Post Checks Written
    3 . Print Transactions                    4 . Print Check Register
    5 . Print Income Statement                6 . Print Balance Sheet
    7 . Access COA file                       8 . Terminate

    MAKE SELECTION  ?
```

Sample 2. P&L statement.

```
**************************************************************************
                         SuperSoftware, Inc.

  INTERIM REPORT
                           INCOME STATEMENT

                            March 1, 1980
```

	CURRENT BALANCE	CURRENT PERCENT	Y-T-D BALANCE	Y-T-D PERCENT
REVENUE FROM SALES				
CASH - SALES	0.00	0.00	100.00	57.14
SALES - ON ACCOUNT	0.00	0.00	75.00	42.85
NET REVENUE FROM SALES	0.00	0.00	175.00	100.00
COST OF SALES				
COST OF MDSE. SOLD	0.00	0.00	0.00	0.00
TOTAL COST OF SALES	0.00	0.00	0.00	0.00
********************				
GROSS PROFIT ON SALES	0.00	0.00	175.00	100.00
********************				
OPERATING EXPENSES				
ADVERTISING & SALES PROM.	0.00	0.00	0.00	0.00
ACCOUNTING & AUDIT FEES	0.00	0.00	0.00	0.00
BANK SERVICE CHARGE	0.00	0.00	0.00	0.00
DATA PROCESSING	0.00	0.00	0.00	0.00
DEPRECIATION	0.00	0.00	0.00	0.00
DONATION	0.00	0.00	0.00	0.00
DUES & SUBSCRIPTION	0.00	0.00	0.00	0.00
FREIGHT	0.00	0.00	5.95	3.40
INSURANCE	0.00	0.00	0.00	0.00
INTEREST	0.00	0.00	0.00	0.00
LEGAL	0.00	0.00	0.00	0.00
MAINTENANCE	0.00	0.00	0.00	0.00
INVENTORY SHORTAGE	0.00	0.00	0.00	0.00
OFFICE SUPPLIES	0.00	0.00	0.00	0.00
PAYROLL	0.00	0.00	0.00	0.00
PAYROLL TAXES	0.00	0.00	0.00	0.00
POSTAGE	0.00	0.00	0.00	0.00
RENT	0.00	0.00	0.00	0.00
TELEPHONE & TELEGRAPH	0.00	0.00	0.00	0.00
TRAVEL & ENTERTAINMENT	0.00	0.00	0.00	0.00
UTILITIES	0.00	0.00	0.00	0.00
TOTAL OPERATING EXPENSE	0.00	0.00	5.95	3.40
********************				
NET OPERATING INCOME	0.00	0.00	169.05	96.60
********************				
OTHER INCOME				
INTEREST INCOME	0.00	0.00	0.00	0.00
RENTAL INCOME	0.00	0.00	0.00	0.00
CASH OVER OR SHORT	0.00	0.00	0.00	0.00
SALE OF ASSETS	0.00	0.00	0.00	0.00
TOTAL - OTHER INCOME	0.00	0.00	0.00	0.00
OTHER EXPENSES				
BAD DEBTS	0.00	0.00	0.00	0.00
PENALITIES	0.00	0.00	0.00	0.00
COST OF ASSETS SOLD	0.00	0.00	0.00	0.00
MISCELLANEOUS EXPENSES	0.00	0.00	0.00	0.00
TOTAL - OTHER EXPENSES	0.00	0.00	0.00	0.00
NET PROFIT (LOSS)	0.00	0.00	169.05	96.60

```
**************************************************************************
```

expansion we might want to make. When we get our payroll program all we need do is remove the colon in line 1030 and our menu will be current.

The INCOME.STM Program

The P&L statement will be printed out on the port number you select. You must tell the program if it is the end of the month during the course of its execution. If it is the end of the month, we store a flag (lines 30–40) so the BALANC.SHT program knows it. If it is not the end of the month, the report will print the word INTERIM in the header. (In the expanded version we might also use this information to automatically update the COA for any depreciable assets we have.)

After certain program variables are obtained (lines 9800–9890), we enter into routine 1000–1090 to get the current period totals and year-to-date totals for all account numbers greater than 399. The remainder of the program prints out, in a nice format, the necessary information for the income statement. The last line of the report prints the net profit or loss as previously defined. Sample 2 shows a typical income statement. This program is shown in Listing 3.

Sample 3. Balance sheet.

```
                    SuperSoftware, Inc.

                       BALANCE SHEET

                       March 1, 1980

ASSETS

    ** CURRENT ASSETS **
CHECKING                                 95094.05
PETTY CASH                                   0.00
PREPAID INSURANCE                            0.00
PREPAID INTEREST                             0.00
INSURANCE - CASH VALUE                       0.00
A/R - TRADE                                 75.00
A/R - EMPLOYEES                              0.00
INVENTORY                                    0.00
 TOTAL CURRENT ASSETS                                95169.05

    ** FIXED ASSETS **
LAND                                         0.00
DEPRECIABLE ASSETS                        5000.00
LESS DEPRECIATION CLAIMED                    0.00
 NET FIXED ASSETS                                     5000.00

    ** OTHER ASSETS **
DEFERRED INTEREST EXPENSE                    0.00
ORGANIZATIONAL EXPENSE                       0.00
 TOTAL OTHER ASSETS                                      0.00


    ************
    TOTAL ASSETS                                                100169.05
    ************


LIABILITIES AND STOCKHOLDERS' EQUITY


    ** CURRENT LIABILITY **
ACCOUNTS PAYABLE                             0.00
FICA WITHHELD                                0.00
FED. TAX WITHHELD                            0.00
STATE TAX WITHHELD                           0.00
STATE - UNEMP TAX                            0.00
FED. UNEMP TAX                               0.00
ACCRUED INTEREST PAYABLE                     0.00
 TOTAL CURRENT LIABILITIES                               0.00

    ** LONG TERM LIABILITIES **
NOTE #1 PAYABLE                              0.00
NOTE #2 PAYABLE                              0.00
 TOTAL LONG TERM LIABILITIES                             0.00


    ** STOCKHOLDERS' EQUITY **
CAPITAL                                 100000.00
RETAINED EARNINGS                          169.05
NET PROFIT                                   0.00
 TOTAL STOCKHOLDERS' EQUITY                         100169.05


    **************************************
    TOTAL LIABILITY & STOCKHOLDERS' EQUITY                      100169.05
    **************************************
BOOKS ARE IN BALANCE. GOOD WORK.
```

The BALANC.SHT Program

This program may only be entered after running INCOME.STM. Following the printout (see Sample 3), a test of the two net profits is made (lines 560–570). If they are not equal, a report of the absolute difference is printed out and CHART.COR is called. If they are equal and it is the end of the month, the current balances of accounts 400 and greater are set to zero. If they are equal you are congratulated and LEDGER.BAS is called. The program is shown in Listing 4.

The TRANS.BAS Program

This will record all transactions, where money is shifted from one COA to another, and save the information in TRANS.DAT. If the transaction involves a check being written, we assume the money will be taken out of account 100, checking. Since amounts can be transferred from any account to any other account, we need a method to determine whether we should add or subtract the amount involved from the second account.

We will always subtract the amount of money involved from the first account (the From Account). Whether to add or subtract to the second account (the To Account) is based on how the net profit will be affected by the amount of money being subtracted from the From Account. The current value of the array D(8,8), which contains only *plus or minus* ones, and the first number of the accounts determine whether to add or to subtract from the To Account. This is accomplished in lines 5000–5090. For proper operation of this program the coding of lines 9900–9924 must be absolutely correct.

You will find that negative dollars will be processed in the same manner as positive dollars. Thus, negative amounts are permitted in any account. This program is shown in Listing 5.

The TRANS.PRT Program

This program will print out a report of all records stored in TRANS.DAT. Following the printout and before returning to LEDGER.BAS, you may reset the pointer to 0. An alternative method of reset is to execute program MASTER.CHG and set the pointer to

FMG CORPORATION

FMG CORPORATION NOW CARRIES GRAHAM-DORIAN & PEACHTREE SOFTWARE

VISA® | master charge

P.O. Box 16020
Fort Worth, Texas
76133
(817) 294-2510

✓70

M-530

All FMG Software Products Include All Necessary Manuals

NEW VERSATILITY
For Your TRS-80

CP/M®2

CONTROL PROGRAM FOR MICROCOMPUTERS ENABLING YOU TO RUN SOFTWARE PUBLISHED FOR CP/M 1.4 ON THE TRS-80

CP/M is considered the industry standard disk operating system because it gives you the hardware-independent interface you need to make your computer work for you. CP/M 2.0 is the *latest in the evolution of a proven* reliable and efficient software system. FMG CORPORATION NOW OFFERS THE CP/M 2.0 FOR THE TRS-80.
It features an enhanced upward compatible file system and powerful new random access capabilities. The CP/M 2.0 from FMG provides the ability to run software published for the CP/M system, on the TRS-80 Model II. From minidisks, floppy disks, all the way to high-capacity hard disks, the flexibility of CP/M 2.0 makes it a truly universal operating system. The package includes an 8" system disk, editor, assembler and debugger for the TRS-80

Available in Format A, B, C, G only . . . $200/$25

MP/M™

MULTI-PROGRAMMING MONITOR

NEW INDUSTRY STANDARD

A deluxe operating system that provides big computer facilities at small computer prices. MP/M is a monitor program which operates with your microcomputer to provide multi-terminal access with multi-programming at each terminal. Best of all, it's CP/M compatible which means you can run a wide variety or programming languages, applications packages, and development software.

You can run simultaneous editors, program translators, and background printer spoolers. Or you can use MP/M for data entry or data-base access from remote terminals. Or you can use MP/M real-time features to monitor an assembly line and automatically schedule programs for execution throughout the day. MP/M makes an excellent focal point for a cluster of connected microcomputers. The possibilities are limitless.

(Format B) . . . . $450/$35
(Format G) . . . . $300/$35

*CP/M and MP/M are trademarks of Digital Research.
Z80 is a trademark of Zilog, Inc.
TRS-80 is a trademark of Tandy Corp.
Pascal/M is a trademark of Sorcim.

· MICROPRO INTERNATIONAL

SOFTWARE WITH MANUAL / MANUAL ALONE

(M) **SUPER-SORT I** — Sort, merge, extract utility as absolute executable program or linkable module in Microsoft format. Sorts fixed or variable records with data in binary, BCD, Packed Decimal, EBCDIC, ASCII, floating & fixed point, exponential, field justified, etc. Even variable number of fields per record! $225/$25

SUPER-SORT II — Above available as absolute program only . . . . $175/$25

SUPER-SORT III — As II without SELECT/EXCLUDE . . . . $125/$25

WORD-STAR — Menu driven visual word processing system for use with standard terminals. Text formatting performed on screen. Facilities for text paginate, page number, justify, center and underscore. User can print one document while simultaneously editing a second. Edit facilities include global search and replace, Read/Write to other text files, block move, etc. Requires CRT terminal with addressable cursor positioning . . . . $495/$40

WORD-STAR Customization Notes — For sophisticated users who do not have one of the many standard terminal or printer configurations in the distribution version of WORD-STAR . . . . NA/$95

WORD-MASTER Text Editor — In one mode has superset of CP/M's ED commands including global searching and replacing, forwards and backwards in file In video mode, provides full screen editor for users with serial addressable-cursor terminal . . . . $150/$25

FLOPPY SAVER — Protection for center holes of 5" and 8" floppy disks. Only 1 needed per diskette. Kit contains centering post, pressure tool and tough 7 mil mylar reinforcing rings for 25 diskettes.
5", Kit $14.95
5", Rings only $7.95
8", Kit $16.95
8", Rings only $8.95

HEAD CLEANING DISKETTE — Cleans the drive Read/Write head in 30 seconds. Diskette absorbs loose oxide particles, fingerprints, and other foreign particles that might hinder the performance of the drive head Lasts at least 3 months with daily use.
8" $32.00
5¼" $30.00

(M) · **DESPOOL** — Allows flexibility and efficiency. (Disk file printing can be accomplished while simultaneously using the computer for other tasks) Slower printers do not tie up the computer. Requires 32K minimum . . . . . . $75/$10

(M) · **SCREEN EDIT** — Text editor for program entry — allows user the ability to see entries as they are being made. Has command which enables user to move the viewed position of the file anywhere within the current data file OR add information anywhere in the file. Requires 16K minimum . . . . $125/$25
(Also available in TRS DOS format.
Specify model or TRS-80)

(M) · **MAC** — Disk-based, powerful macro assembler utilizes Standard Intel Mnemonics. Includes macro processor.
The CP/M 8080 Macro Assembler reads assembly language statement from a diskette file and produces an Intel "HEX" format object file on the disk suitable for processing in the TRS-CP/M environment. Requires 32K minimum and CP/M . . . . $100/$25

(M) · **ZSID** — Efficient and reliable program testing system for Z80 microcomputers. Capabilities include traceback and histogram facilities. Allows real time break points.
ZSID is a symbolic debugger which expands upon the features of the TRS-CP/M standard debugger, providing greatly enhanced facilities for assembly language program check-out. Requires 32K minimum and CP/M . . . . . . $99/$25

(M) **MAIL LIST** — Mailing list maintenance package. No sorting required to print normal address labels in zip code sequence. Supports new larger zip code. Sorts and selects on multiple fields. Labels may be printed in user selectable formats. Includes sort and select utilities $300/$35

· PEACHTREE SOFTWARE SYSTEMS

(T) **GENERAL LEDGER** — Records details of all financial transactions. Generates a balance sheet and an Income statement. Flexible and adaptable design for both small businesses and firms performing client writeup services. Produces reports as follows: Trial Balance, Transaction Registers, Balance Sheet, Prior Year Comparative Balance Sheet, Income Statement, Prior Year Comparative Income Statement and Department Income Statements. Interactive with other PEACHTREE accounting packages. Supplied in source code for Microsoft BASIC . . . . . . $990/$30

(T) **ACCOUNTS PAYABLE** — Tracks current and aged payables and incorporates a check writing feature. Maintains a complete vendor file with information on purchase orders and discount terms as well as active account status. Produces reports as follows: Open Voucher Report, Accounts Payable Aging Report and Cash Requirements. Provides input to PEACHTREE General Ledger. Supplied in source code for Microsoft. BASIC . . . . $990/$30

(T) **ACCOUNTS RECEIVABLE** — Generates invoice register and complete monthly statements. Tracks current and aged receivables. Maintains customer file including credit information and account status. The current status of any customer account is instantly available. Produces reports as follows: Aged Accounts Receivable, Invoice Register, Payment and Adjustment Register and Customer Account Status Report. Provides input to PEACHTREE General Ledger. Supplied in source code for Microsoft BASIC . . $990/$30

(T) **PAYROLL** — Prepares payroll for hourly, salaried and commissioned employees. Generates monthly, quarterly and annual returns. Prepares employee W-2's. Includes tables for federal withholding and FICA as well as withholding for all 50 states plus up to 20 cities from pre-computed or user generated tables. Will print checks, Payroll Register, Monthly Summary and Unemployment Tax Report. Provides input to PEACHTREE General Ledger. Supplied in source code for Microsoft BASIC . . . . $990/$30

(T) **INVENTORY** — Maintains detailed information on each inventory item including part number, description, unit of measure, vendor and reorder data, item activity and complete information on current item costs, pricing and sales. Produces reports as follows: Physical Inventory Worksheet, Inventory Price List, Departmental Summary Report, Inventory Status Report, The Reorder Report and the Period-to-Date and Year-to-Date reports. Supplied in source code for Microsoft BASIC . . . . $1,190/$30

(T) **MAILING ADDRESS** — Keeps track of name and address information and allows the selective printing of this information in the form of mailing lists or address labels. Allows the user to tailor the system to his own particular requirements. User-defined format and print-out system uses a special format file which tells programs how to print the mailing list or address labels. Standard format files are included with system. Automatic sorting of data uses indexed file management routines which allow the name and address information to be sequentially retrieved and printed without file sorting Supplied in source code for Microsoft BASIC . . . . $790/$30

· GRAHAM-DORIAN SOFTWARE SYSTEMS

(T) **GENERAL LEDGER** — An on-line system; no batching is required. Entries to other GRAHAM-DORIAN accounting packages are automatically posted. User establishes customized C.O.A. Provides transaction register, record of journal entries, trial balances and monthly closings. Keeps 14 month history and provides comparison of current year with previous year. Requires CBASIC-2. Supplied in source . . $995/$35

(T) **ACCOUNTS PAYABLE** — Maintains vendor list and check register. Performs cash flow analysis. Flexible — writes checks to specific vendor for certain invoices or can make partial payments. Automatically posts to GRAHAM-DORIAN General Ledger or runs as stand alone system. Requires CBASIC-2. Supplied in source . . . . $995/$35

(T) **ACCOUNTS RECEIVABLE** — Creates trial balance reports, prepares statements, ages accounts and records invoices Provides complete information describing customer payment activity. Receipts can be posted to different ledger accounts. Entries automatically update GRAHAM-DORIAN General Ledger or runs as stand alone system. Requires CBASIC-2 Supplied in source . . . . $995/$35

(T) **PAYROLL SYSTEM** — Maintains employee master file. Computes payroll withholding for FICA, Federal and State taxes. Prints payroll register, checks, quarterly reports and W-2 forms. Can generate ad hoc reports and employee form letters with mail labels. Requires CBASIC-2. Supplied in source . . . . . . $590/$35

(T) **INVENTORY SYSTEM** — Captures stock levels, costs, sources, sales, ages, turnover, markup, etc. Transaction information may be entered for reporting by salesman, type of sale, date of sale, etc. Reports available both for accounting and decision making. Requires CBASIC-2. Supplied in source . $590/$35

(T) **JOB COSTING** — Designed for general contractors. To be used interactively with other GRAHAM-DORIAN accounting packages for tracking and analysing expenses User establishes customized cost categories and job phases. Permits comparison of actual versus estimated costs Automatically updates GRAHAM-DORIAN General Ledger or runs as stand alone system. Requires CBASIC-2 Supplied in source $995/$35

Sample Program Disk For Each Graham-Dorian Business Package. Specify Package $45

The sale of each proprietary software package conveys a license for use on one system only.

Prices F.O.B. Fort Worth, Tex. Shipping, handling and C.O.D. charges extra.

· MICROSOFT PRODUCTS

(M) **BASIC-80** — Disk Extended BASIC, ANSI compatible with long variable names, WHILE/WEND, chaining, variable length file records . . . . $350/$25

(M) **BASIC COMPILER** — Language compatible with BASIC-80 and 3-10 times faster execution. Produces standard Microsoft relocatable binary output. Includes MACRO-80. Also linkable to FORTRAN-80 or COBOL-80 code modules . . . . $395/$25

(M) **FORTRAN-80** — ANSI 66 (except for COMPLEX) plus many extensions. Includes relocatable object compiler, linking loader, library with manager. Also includes MACRO-80 *(see below)* . . . . $500/$25

(M) **COBOL-80** — Level 1 ANSI '74 standard COBOL plus most of Level 2. Full sequential, relative, and indexed file support with variable file names. STRING, UNSTRING, COMPUTE, VARYING/UNTIL, EXTEND, CALL, COPY, SEARCH, 3-dimensional arrays, compound and abbreviated conditions, nested IF. Powerful interactive screen-handling extensions Includes compatible assembler, linking loader, and relocatable library manager as described under MACRO-80 . . . . $750/$25

(M) **MACRO-80** — 8080/Z80 Macro Assembler. Intel and Zilog mnemonics supported. Relocatable linkable output. Loader, Library Manager and Cross Reference List utilities included . . . . . . $150/$25

(M) **XMACRO-86** — 8086 cross assembler. All Macro and utility features of MACRO-80 package. Mnemonics slightly modified from Intel ASM86. Compatibility data sheet available . . . . $300/$25

(M) **PASCAL/M*** — Compiler generates P code from extended language, implementation of standard PASCAL. Supports overlay structure through additional procedure calls and the SEGMENT procedure type. Provides convenient string handling capability with the added variable type STRING. Untyped files allow memory Image I/O. Requires 56K CP/M . . . $150/$20

(M) **PASCAL/Z** — Z80 native code PASCAL compiler. Produces optimized, ROMable re-entrant code. All interfacing to CP/M is through the support library. The package includes compiler, Microsoft Compatible relocating assembler and linker, and source for all library modules Variant records, strings and direct I/O are supported. Requires 56K CP/M and Z80 CPU. . . . . $395/$25

(M) **PASCAL/MT** — Subset of standard PASCAL. Generates ROMable 8080 machine code. Symbolic debugger included Supports interrupt procedures, CP/M file I/O and assembly language Interface. Real variables can be BCD, software floating point, or AMD 9511 hardware floating point. Version 3 includes Enumeration and Record data types. Manual explains BASIC to PASCAL conversion. Source for the run-time package requires Digital Research's MAC. Requires 32K . . . . $250/$30

(M) **CBASIC-2** Disk Extended BASIC — Non-interactive BASIC with pseudo-code compiler and run-time interpreter. Supports full file control, chaining. integer and extended precision variables, etc. . . $110/$15

(M) **BSTAM** — Utility to link one computer to another also equipped with BSTAM. Allows file transfers at full data speed (no conversion to hex), with CRC block control check for very reliable error detection and automatic retry. We use it! It's great! Full wildcard expansion to send ★ COM, etc. 9600 baud with wire. 300 baud with phone connection. Both ends need one. Standard and (M) versions can talk to one another. . . . . $150/$5

(M) (T) **SELECTOR III-C2** — Data Base Processor to create and maintain multi Key data bases. Prints formatted sorted reports with numerical summaries or mailing labels Comes with sample applications, including Sales Activity, Inventory, Payables, Receivables, Check Register, and Client/Patient Appointments, etc. Requires CBASIC-2. Supplied in source . $349/$20

GLECTOR — General Ledger option to SELECTOR III-C2. Interactive system provides for customized COA. Unique chart of transaction types insure proper double entry bookkeeping Generates balance sheets, P&L statements and *journals*. Two year record allows for statement of changes in financial position report. Supplied in source. Requires SELECTOR III-C2, CBASIC-2 and 52K system . . . . $250/$25

(M) **TEXTWRITER III** — Text formatter to justify and paginate letters and other documents. Special features Include insertion of text during execution from other disk files or console, permitting recipe documents to be created from linked fragments on other files. Has facilities for sorted index, table of contents and footnote insertions. Ideal for contracts, manuals, etc. Now compatible with Electric Pencil* prepared files. . . . . $125/$20

ORDERS MUST SPECIFY DISK SYSTEMS AND FORMATS:

FORMATS AVAILABLE:
(A) TRS-80 Model I (M) Keys Only
(B) TRS-80 Model II
(C) TRS-80 Model III (M) Keys Only
(D) HEATHKIT H89 (M) Keys Only
(E) NORTH STAR
(F) SUPER BRAIN QD
(G) STANDARD UNIMPLEMENTED

(M) Modified version available for use with CPM as implemented on Heath and TRS-80 Model I computers.

(T) For all (T) items listed above, the recommended system configuration consists of 48K CPM 2 full dize disk drives. 24 x 80 CRT and 132 column printer.

FMG's LIBRARY:

PASCAL USER MANUAL & REPORT
(2nd) Edition by K. Jensen and N. Wirth

- **Tutorial Manual and Concise Referenca Report for Both Programmers and Implementors**
- **Includes Haipful Exampias to Demonstrate tha Various Faatures of PASCAL**

The book consists of two parts the user manual and the revised report. The manual is directed to those who have some familiarity with computer programming and who wish to get acquainted with the PASCAL language The report defines standard PASCAL, which constitutes a common base between various implementations of the language

Stock No. #B21 — Prica $9.95

PASCAL PRIMER Problem Solving

This book has three major goals:
- **To Introduce all aspects of the programming and problem solving process (Includes problem specification and organization, algorithms, coding, debugging, testing, documentation and maintenance);**
- **To teach good programming style and how to produce a high quality finished product; and**
- **To taach the syntax of tha PASCAL programming language.**

Numerous examples are employed throughout the text PASCAL is used as a vehicle to teach various aspects of programming techniques

Stock No. #B24 — Price $18.95

Microcomputer Problem Solving Using Pascal
by Kenneth L. Bowles

- **A Book Dasigned for Both College Courses AND Individual Self-Study**
- **Idaal for use with UCSD Pascal**
- **Includes Extensions to Standard PASCAL**

This book is designed both for introductory courses in computer problem solving at the freshman and sophomore college level, and for individual self-study it includes many examples and actually executable programs It includes information on the necessary functions and procedures for handling graphics and strings

Stock No. #B22 — Prica $14.95

BEGINNER'S MANUAL FOR UCSD PASCAL SYSTEM

- **An Enlightening introduction to UCSD PASCAL**
- **Demonstrates How to Use the UCSD PASCAL System and How to Program In PASCAL**
- **Includes Many Practical Examples of PASCAL Programs**

This book is intended to be used as an introduction and reference manual for people just beginning to use the UCSD Pascal Software System Whether you have never used a computer before or whether you are an experienced programmer who is unfamiliar with UCSD PASCAL, this book will provide a relatively easy, yet thorough, introduction to UCSD PASCAL

Stock No. #B25 — Price $11.95

PROGRAMING IN PASCAL
by Peter Grogono

- **An Excellent Introduction to One of the Fastest Growing Programming Languages Today**
- **Sections on Procadures and Files PLUS a Chapter on Dynamic Data Structuras such as Trees and Linkad Lists**

The text is arranged as a tutorial, containing both examples and exercises to increase reader proficiency in PASCAL. Concepts are illustrated by examples, ranging from the Tower of Hanoi problem to circumscribing a circle about a triangle PROGRAMMING IN PASCAL is sure to hold the reader's interest.

Stock No. #B23 — Price $14.95

UCSD Reference Book

- **A Refaranca Guide to tha Complete UCSD PASCAL Systam**
- **Includes Information on Compiler Basic, Assembiar and Editor**
- **Lists Actual P-Machine Codes**

This reference book can be a valuable and time-saving guide to thorough information on the UCSD PASCAL system The easy-to-read manual provides fast access to pertinent data

Stock No. #B26 — Price $25.00

✓ Reader Service—see page 226

0 manually (record number 6). This program is shown in Listing 6.

Expanding the System

Several items will immediately enhance this system. For instance, if during the saving of each transaction we also saved in another file the date and record number of each transaction, we could have a report of each transaction sorted by date. We could also save each account number of each transaction, and then sort by account number. This would give us a report of transactions by account number. TRANS.PRT could be expanded to print only checks and/or intra-chart transactions.

The addition of a payroll program which would update all accounts would save lots of time. We would also like to know what our federal and state tax liabilities are. If we owe payroll taxes, then a program to tell us which forms to use, when they are due and all other information would be of great value. If our business is going to include accounts receivable, accounts payable and inventory, we must, of course, add those at some time in the future.■

Sample 4. Transaction file.

```
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
                          SuperSoftware, Inc.

                   Transactions by ORDER ENTERED

TODAY'S DATE:                March 1, 1980

= = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
  DATE        ACCOUNT #          DESCRIPTION                 AMOUNT      CK. #
= = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
 1. 1/1/80        300         CAPITAL                      -50000.00
                  100         CHECKING
PURPOSE: Sold stock

 2. 1/1/80        300         CAPITAL                      -50000.00
                  100         CHECKING
PURPOSE: Sold more stock to you.

 3. 1/2/80        100         CHECKING                       5000.00       100
                  160         DEPRECIABLE ASSETS
PURPOSE: Bought 6800 cpu system

 4. 1/3/80        100         CHECKING                          5.95       101
                  617         FREIGHT
PURPOSE: Freight UPS-cpu system

 5. 1/20/80       400         CASH - SALES                   -100.00
                  100         CHECKING
PURPOSE: Sold First Program

 6. 1/22/80       401         SALES - ON ACCOUNT              -75.00
                  110         A/R - TRADE
PURPOSE: Sold on account 2nd pgm
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
```

Listing 1. BEGIN program.

```
0001 : BEGIN
0002 :
0005 : Embry  3/1/80
0006 :
0007 : Kb article
0008 :
0010 HOME
0014 PRINT "Hello!"
0016 PRINT
0018 PRINT "Ready to have another go at it, I see."
0020 PRINT
0099 :
0100 : Inputs
0101 :
0110 PRINT "Input Today's Date"
0112 PRINT
0120 INPUT "Month ",M
0130 IF M<1 THEN PRINT CHR$(7):GOTO 110
0140 IF M>12 THEN PRINT CHR$(7):GOTO 110
0150 LET M$=STR$(M)
0160 IF LEN(M$)<2 THEN M$="0"+M$
0199 :
0200 INPUT "Day ",D
0210 IF D<1 THEN PRINT CHR$(7):GOTO 200
0220 IF D>31 THEN PRINT CHR$(7):GOTO 200
0230 LET D$=STR$(D)
0240 IF LEN(D$)<2 THEN D$="0"+D$
0299 :
0300 INPUT "Year ",Y
0301 :
0310 IF Y<0 THEN PRINT CHR$(7):GOTO 300
0320 IF Y<100 THEN Y=1900+Y
0330 LET Y$=STR$(Y)
0340 IF LEN(Y$)<>4 THEN 300
0399 :
0400 : FORMAT DATE
0401 :
0410 LET T$=Y$+M$+D$::Format date in decreasing order  (YYYYMMDD)
0420 LET T=VAL(T$)
0430 LET D=VAL(D$):D$=STR$(D)::Get rid of leading 0's
0499 :
0500 : JULIAN DAY
0501 :
0510 LET J=D-32075+1461*(Y+4800+(M-14)/12)/4
0520 LET J=J+367*(M-2-(M-14)/12*12)/12
0530 LET J=J-3*((Y+4900+(M-14)/12)/100)/4
0540 LET J=INT(J)
0599 :
0600 : Save it
```

More

Listing 1 continued.

```
0601 :
0610 OPEN #19,0:MASTER.DAT
0620 RECNO #19=2:GOSUB 9000
0630 LET A=T:B=J
0640 GOSUB 9100::Put Julian Day Here - might need it.
0650 RECNO #19=M:GOSUB 9000::Get this month
0660 DLM=OFF
0670 LET T$=A$+" "+D$+","+" "+Y$
0680 RECNO #19=14:GOSUB 9000
0682 LET A$=T$
0684 GOSUB 9100::Put the Readable date here!
0690 DLM=ON
0699 :
0900 : END
0901 :
0990 CHAIN LEDGER.BAS
0999 :
9000 : GET   ->  MASTER.DAT
9001 :
9010 GET #19,A,B,A$
9090 RETURN
9099 :
9100 : PUT    ->  MASTER.DAT
9101 :
9110 PUT #19,A,B,A$
9190 RETURN
```

Listing 2. LEDGER.BAS program.

```
0001 : LEDGER.BAS
0002 :
0003 : Gene Embry 4/1/80
0004 :
0005 : Kilobaud Demonstration
0006 :
0009 :
0010 LET W=35
0012 DLM=OFF
0020 LINE= 0
0030 OPEN #19,0:MASTER.DAT
0032 RECNO #19=13:READ #19,A,B,H$::Name
0034 READ #19,A,B,D$::Today-Readable
0040 CLOSE #19
0050 LET L$="Minimum Accounting System"
0082 DLM=ON
0099 :
0100 : INTRO
0101 :
0110 HOME
0112 PRINT TAB(W-LEN(H$)/2);H$
0114 PRINT TAB(W-LEN(D$)/2);D$
0120 PRINT
0126 PRINT TAB(W-LEN(L$)/2);L$
0128 PRINT
0130 READ N$,P$
0132 IF N$="0" THEN 140
0134 LET T=T+1
0136 GOTO 130
0140 DIM X$(T),P$(T)
0142 RESTORE
0144 FOR X=1 TO T:READ X$(X),P$(X):NEXT X
0150 FOR X=1 TO T STEP 2
0152 IF X+1>T P.X;". ";X$(X):GOTO 160
0154 PRINT X;". ";X$(X);TAB(W+5);X+1;". ";X$(X+1)
0156 NEXT X
0160 PRINT
0162 INPUT "MAKE SELECTION ",S
0199 :
0200 : DO SOMETHING
0201 :
0210 IF S>T-1 THEN 900
0220 IF S<1 THEN 990
0230 IF X$(S)="Print Balance Sheet" P."First we must do the P&L Statement."
0290 CHAIN P$(S)
0899 :
0900 :
0910 PRINT
0920 PRINT "Bye!"
0990 END
0999 :
1000 DATA Post Transactions,TRANS.BAS
1010 DATA Post Checks Written,TRANS.BAS
1020 DATA Print Transactions,TRANS.PRT
1025 DATA Print Check Register,TRANS.PRT
1030 : DATA Access Payroll Files,MAIN.PAY
1032 : DATA Print Payroll,PAYROL.BAS
1040 : DATA Access A/R Files,MAIN.AR
1042 : DATA Access A/P Files,MAIN.AP
1050 : DATA Access Inventory Files,INVENT.BAS
1060 DATA Print Income Statement,INCOME.STM
1070 DATA Print Balance Sheet,INCOME.STM
```

More →

Listing 2 continued.

```
1072 DATA Access COA file,CHART.COR
1080 DATA Terminate,0
1090 DATA 0,0
```

Listing 3. INCOME.STM program.

```
0001 : INCOME.STM
0002 :
0003 : Embry  3/1/80
0004 :
0005 : Kb article
0006 :
0007 : Corporation
0008 :
0010 HOME
0012 PRINT "Income Statement"
0020 GOSUB 9800::Get program variables
0030 INPUT "Is this the end of the period (Y/N)  ",T$
0032 IF T$<>"Y" IF T$<>"N" THEN 30
0034 IF T$="N" THEN B1=0::Not end of month
0036 IF T$="Y" THEN B1=1::end of month
0038 GOSUB 8200::Save flag
0050 INPUT "Which port for output  ",Q
0060 PRINT "GETTING DATA"
0080 GOSUB 1000::Get ready
0099 :
0200 : PRINT HEADING
0201 :
0205 IF Q=1 THEN HOME
0210 LET H$="**":GOSUB 9100
0212 PRINT #Q,TAB(W-LEN(N$)/2);N$:P.#Q
0214 IF T$<>"Y" P.#Q,J$
0216 PRINT #Q,TAB(W-LEN(E$)/2);E$
0217 PRINT #Q
0218 PRINT #Q,TAB(W-LEN(D$)/2);D$
0220 SKIP #Q,2
0222 LET H$="- ":GOSUB 9100
0230 PRINT #Q,TAB(W1);F$;
0232 PRINT #Q,TAB(W2);F$;TAB(W3);G$;
0234 PRINT #Q,TAB(W4);G$
0240 PRINT #Q,TAB(W1);I$;TAB(W2);K$;
0242 PRINT #Q,TAB(W3);I$;TAB(W4);K$
0280 GOSUB 9100
0299 :
0300 : WORK AREA
0301 :
0310 DIGITS= 2:RJUST=5
0320 FOR Z=1 TO 5
0330 PRINT #Q
0340 ON Z GOSUB 2000,2100,3000,4000,5000
0350 PRINT #Q
0360 NEXT Z
0399 :
0400 : ENDING
0401 :
0410 SKIP #Q,2
0420 PRINT #Q,"NET PROFIT (LOSS)";
0425 LET Z=T0+T3-T1-T2-T4
0427 LET Z0=T5+T8-T6-T7-T9
0430 PRINT #Q,TAB(W1);Z;
0432 LET N3=T0:F=Z:W9=W2:GOSUB 9200
0436 PRINT #Q,TAB(W3);Z0;
0438 LET N3=T5:F=Z0:W9=W4:GOSUB 9200
0440 PRINT #Q
0490 LET H$="**":GOSUB 9100
0499 :
0500 : CLOSING
0501 :
0580 IF Q=1 THEN 800
0590 SKIP #Q,15
0599 :
0800 : Ending Procedure
0801 :
0810 DIGITS= 0:RJUST=0
0820 RECNO #19=7:GOSUB 9020:A=Z:GOSUB 9030::PROFIT
0899 :
0900 : DONE
0901 :
0910 DLM=ON
0990 CHAIN BALANC.SHT
0999 :
1000 : GET TOTALS
1001 :
1010 FOR X=1 TO T
1020 RECNO #10=X:GOSUB 9010
1022 IF A<400 THEN 1060
1030 IF F1=0 IF INT(A)>=400 THEN Z1=X:F1=1
1048 IF A>=400 IF A<=499 THEN T0=T0+B:T5=T5+C::Sales
1049 IF A>=500 IF A<=599 THEN T1=T1+B:T6=T6+C::Cost of Mdse. Sold
1050 IF A>=600 IF A<=699 THEN T2=T2+B:T7=T7+C::Expenses
1052 IF A>=700 IF A<=799 THEN T3=T3+B:T8=T8+C::Other income
1054 IF A>=800 IF A<=899 THEN T4=T4+B:T9=T9+C::Other Expenses
```

More →

Listing 3 continued.

```
1060 NEXT X
1090 RETURN
1099 :
2000 : OPERATING INCOME  Z=1
2001 :
2010 LET L=400:U=499
2012 LET N1=T0:N2=T5
2014 LET L$="REVENUE FROM SALES"
2016 PRINT #Q,TAB(W-LEN(L$)/2);L$
2020 GOSUB 8000::FIND & PRINT ACCTS.
2040 PRINT #Q,TAB(3);"NET REVENUE FROM SALES";
2050 PRINT #Q,TAB(W1);T0;
2052 LET N3=T0:F=T0:W9=W2
2054 GOSUB 9200
2060 PRINT #Q,TAB(W3);T5;
2062 LET N3=T5:F=T5:W9=W4
2064 GOSUB 9200
2070 PRINT #Q
2090 RETURN
2099 :
2100 : COST Z=2
2101 :
2110 LET L$="COST OF SALES"
2112 PRINT #Q,TAB(W-LEN(L$)/2);L$
2120 LET L=500:U=599
2122 LET N1=T1:N2=T6
2130 GOSUB 8000
2140 PRINT #Q,TAB(3);"TOTAL "+L$;
2142 PRINT #Q,TAB(W1);T1;
2144 LET N3=T0:F=T1:W9=W2:GOSUB 9200
2146 PRINT #Q,TAB(W3);T6;
2148 LET N3=T5:F=T6:W9=W4:GOSUB 9200
2150 PRINT #Q
2152 PRINT #Q
2160 PRINT #Q,TAB(3);"**********************"
2170 PRINT #Q,TAB(3);"GROSS PROFIT ON SALES";
2172 PRINT #Q,TAB(W1);T0-T1;
2174 LET N3=T0:F=T0-T1:W9=W2:GOSUB 9200
2176 PRINT #Q,TAB(W3);T5-T6;
2178 LET N3=T5:F=T5-T6:W9=W4:GOSUB 9200
2186 PRINT #Q,TAB(3);"**********************"
2188 PRINT #Q
2190 RETURN
2199 :
2999 :
3000 : OPERATING EXPENSES  Z=3
3001 :
3010 LET L=600:U=699
3012 LET N1=T2:N2=T7
3020 LET L$="OPERATING EXPENSES"
3022 PRINT #Q,TAB(W-LEN(L$)/2);L$
3030 GOSUB 8000
3040 PRINT #Q,TAB(3);"TOTAL OPERATING EXPENSE";
3042 PRINT #Q,TAB(W1);T2;
3044 LET N3=T0:F=T2:W9=W2:GOSUB 9200
3046 PRINT #Q,TAB(W3);T7;
3048 LET N3=T5:F=T7:W9=W4:GOSUB 9200
3062 PRINT #Q
3064 PRINT #Q
3068 PRINT #Q,"**********************"
3070 PRINT #Q,"NET OPERATING INCOME";
3072 PRINT #Q,TAB(W1);T0-T1-T2;
3074 LET N3=T0:F=T0-T1-T2:W9=W2:GOSUB 9200
3076 PRINT #Q,TAB(W3);T5-T6-T7;
3078 LET N3=T5:F=T5-T6-T7:W9=W4:GOSUB 9200
3080 PRINT #Q
3087 PRINT #Q,"**********************"
3090 RETURN
3099 :
3999 :
4000 : OTHER INCOME  Z=4
4001 :
4010 LET L=700:U=799
4012 LET N1=T3:N2=T8
4020 LET L$="OTHER INCOME"
4022 PRINT #Q,TAB(W-LEN(L$)/2);L$
4030 GOSUB 8000
4060 PRINT #Q,TAB(3);"TOTAL - OTHER INCOME";
4062 PRINT #Q,TAB(W1);T3;
4064 LET N3=T0:F=T3:W9=W2:GOSUB 9200
4066 PRINT #Q,TAB(W3);T8;
4068 LET N3=T5:F=T8:W9=W4:GOSUB 9200
4090 RETURN
4099 :
5000 : OTHER EXPENSES  Z=5
5001 :
5010 LET L=800:U=899
5012 LET N1=T4:N2=T9
5020 LET L$="OTHER EXPENSES"
5022 PRINT #Q,TAB(W-LEN(L$)/2);L$
5030 GOSUB 8000
5060 PRINT #Q,TAB(3);"TOTAL - OTHER EXPENSES";
5062 PRINT #Q,TAB(W1);T4;
5064 LET N3=T0:F=T4:W9=W2:GOSUB 9200
5066 PRINT #Q,TAB(W3);T9;
5068 LET N3=T5:F=T9:W9=W4:GOSUB 9200
5090 RETURN
```

More →

✓ Reader Service—see page 226

Listing 3 continued.

```
5099 :
5999 :
8000 : SELECT DATA FROM FILE
8001 :
8010 FOR X=Z1 TO T
8012 RECNO #10=X:GOSUB 9010
8020 IF A<L THEN 8080
8030 IF A>U THEN Z1=X-1:GOTO 8090
8040 PRINT #Q,A$;
8042 LET F=B
8044 PRINT #Q,TAB(W1);F;
8046 LET N3=T0:W9=W2:GOSUB 9200
8048 LET F=C
8050 PRINT #Q,TAB(W3);F;
8054 LET N3=T5:W9=W4:GOSUB 9200
8070 PRINT #Q
8080 NEXT X
8090 RETURN
8099 :
8200 : END OF MONTH
8201 :
8210 RECNO #19=7
8212 GOSUB 9020::Get
8214 LET B=B1
8216 GOSUB 9030::Put
8290 RETURN
8299 :
8400 : BINARY SEARCH OF CHART.DAT
8401 :
8405 : N=ACCT # FOR SEARCH
8420 LET L=1:U=T
8430 LET I=INT((L+U)/2)
8432 SET RECNO #10=I:GOSUB 9010
8440 IF N=A THEN 8490
8450 IF N<A THEN U=I-1:GOTO 8430
8460 IF N>A THEN L=I+1:GOTO 8430
8490 RETURN
8499 :
9000 : PUT CHART.DAT
9001 :
9004 PUT #10,A,A$,B,C,D
9008 RETURN
9009 :
9010 : GET FROM CHART.DAT
9011 :
9012 GET #10,A,A$,B,C,D
9018 RETURN
9019 :
9020 : GET   ->  MASTER.DAT
9021 :
9022 GET #19,A,B,A$
9028 RETURN
9029 :
9030 : PUT   ->  MASTER.DAT
9031 :
9032 PUT #19,A,B,A$
9038 RETURN
9039 :
9100 : PRINT UNDERSCORE
9101 :
9110 FOR X=1 TO W
9120 PRINT #Q,H$;
9130 NEXT X
9180 PRINT #Q
9190 RETURN
9199 :
9200 : PRINT %
9201 :
9210 IF N3=0 THEN Y=0:GOTO 9240
9215 IF F=0 THEN Y=0:GOTO 9240
9220 LET Y=(F/N3)*100
9225 IF Y<-.01 THEN 9240
9230 IF Y<.01 P.#Q,TAB(W9+2);"< 0.01 ";:GOTO 9290
9240 PRINT #Q,TAB(W9);Y;
9290 RETURN
9299 :
9400 : PRINT A RECORD
9401 :
9410 PRINT #Q,A$,TAB(W1);A;TAB(W3);B
9490 RETURN
9499 :
9800 : PGM VARIABLES
9801 :
9810 LINE= 0
9812 LET W=37::HALF WIDTH OF OUTPUT DEVICE
9814 LET W1=33:W2=44:W3=55:W4=66
9816 DLM=OFF
9820 OPEN #10,0:CHART.DAT
9822 OPEN #19,0:MASTER.DAT
9824 RECNO #19=14:GOSUB 9020
9826 LET D$=A$::TODAY
9830 LET E$="INCOME STATEMENT"
9832 LET F$="CURRENT"
9834 LET G$="Y-T-D"
9836 LET I$="BALANCE"
9838 LET J$="INTERIM REPORT"
```

More

✓Reader Service—see page 226

Listing 3 continued.

```
9840 LET K$="PERCENT"
9842 RECNO #19=13:GOSUB9020:N$=A$::Name
9850 LET T=RSIZE #10
9890 RETURN
```

Listing 4. BALANC.SHT program.

```
0001 : BALANC.SHT
0002 :
0003 : Embry  3/1/80
0004 :
0005 : Kb article
0006 :
0007 : Corporation
0009 :
0010 HOME
0020 LET W=21:LINE=0
0060 PRINT TAB(W);"AND NOW -   THE BALANCE SHEET"
0062 INPUT "Which Port for output  ",Q
0070 GOSUB 1000::GET DATA
0099 :
0100 : INTRODUCTION
0101 :
0110 DLM=OFF
0120 RECNO #19=13:GOSUB 9020:N$=A$::NAME
0130 RECNO #19=14:GOSUB 9020:D$=A$::DATE
0199 :
0200 : PRINT REPORT
0201 :
0205 DIGITS= 2:RJUST=6
0210 IF Q=1 THEN HOME
0212 PRINT #Q,TAB(37-LEN(N$)/2);N$
0214 SKIP #Q,2
0215 LET E$="BALANCE SHEET"
0216 PRINT #Q,TAB(37-LEN(E$)/2);E$
0217 PRINT #Q
0218 PRINT #Q,TAB(37-LEN(D$)/2);D$
0220 SKIP #Q,2
0230 PRINT #Q,"ASSETS"
0232 PRINT #Q
0240 PRINT #Q,TAB(W/4);"** CURRENT ASSETS **"
0250 GOSUB 2000
0260 PRINT #Q
0262 PRINT #Q,TAB(W/4);"** FIXED ASSETS **"
0280 GOSUB 3000
0282 PRINT #Q
0290 PRINT #Q,TAB(W/4);"** OTHER ASSETS **"
0300 GOSUB 4000
0320 SKIP #Q,2
0325 LET M=J+K+L:Z=M
0328 PRINT #Q,TAB(W/4);"************"
0330 PRINT #Q,TAB(W/4);"TOTAL ASSETS";TAB(3*W);M
0332 PRINT #Q,TAB(W/4);"************"
0399 :
0400 : LIABILITIES & EQUITY
0401 :
0410 SKIP #Q,2
0420 PRINT #Q,"LIABILITIES AND STOCKHOLDERS' EQUITY"
0430 SKIP #Q,2
0440 PRINT #Q,TAB(W/4);"** CURRENT LIABILITY **"
0450 GOSUB 5000
0460 PRINT #Q
0470 PRINT #Q,TAB(W/4);"** LONG TERM LIABILITIES **"
0480 GOSUB 6000
0490 SKIP #Q,2
0500 PRINT #Q,TAB(W/4);"** STOCKHOLDERS' EQUITY **"
0510 GOSUB 8000
0515 LET R=N+P+Q1
0517 SKIP #Q,2:P.#Q,TAB(W/4);
0518 PRINT #Q,"****************************************"
0519 PRINT #Q,TAB(W/4);
0520 PRINT #Q,"TOTAL LIABILITY & STOCKHOLDERS' EQUITY";TAB(3*W);R
0521 PRINT #Q,TAB(W/4);
0522 PRINT #Q,"****************************************"
0527 DIGITS= 0:RJUST=0
0540 IF G$<>"Y" THEN 560
0560 IF V7=Z P.#Q,"BOOKS ARE IN BALANCE. GOOD WORK."
0570 IF V7<>Z THEN 9300::OUT OF BALANCE
0580 PRINT #Q,CHR$(12)
0599 :
0600 : ALL DONE
0601 :
0610 IF VG=0 THEN 900
0620 LET N1=310:GOSUB 1200::RETAINED EARNINGS
0630 LET C=C+Z:GOSUB 9000
0640 FOR X=T+1 TO RSIZE #10
0642 RECNO #10=X:GOSUB 9010::set
0644 LET B=0::reset 'current balance' to 0
0646 GOSUB 9000::PUT
0648 NEXT X
0699 :
```

More

Listing 4 continued.

```
0900 : DONE
0901 :
0980 DLM=ON
0990 CHAIN LEDGER.BAS
0999 :
1000 : OPEN AND GET ARRAY
1001 :
1005 PRINT "Getting Data"
1010 OPEN #19,0:MASTER.DAT
1012 OPEN #10,0:CHART.DAT
1018 LET T=RSIZE #10
1020 FOR X=1 TO T
1022 RECNO #10=X:GOSUB 9010
1024 IF A >=400 THEN T=X-1:GOTO 1030
1026 NEXT X
1030 RECNO #19=7:GOSUB 9020:V7=A:V6=B
1090 RETURN
1099 :
1200 : BINARY SEARCH OF CHART.DAT
1201 :
1210 LET L1=1:U=T
1220 LET I=INT((L1+U)/2)
1222 RECNO #10=I:GOSUB 9010::GET
1230 IF N1=A THEN 1290
1240 IF N1<A THEN U=I-1:GOTO 1220
1250 IF N1>A THEN L1=I+1:GOTO 1220
1290 RETURN
1299 :
2000 : CURRENT ASSETS  100 -> 149
2001 :
2010 FOR X=1 TO T
2012 RECNO #10=X:GOSUB 9010::Get
2020 IF A>149 THEN 2081
2022 IF F=1 THEN 2080
2030 IF A<100 THEN 2080
2040 LET J=J+C
2045 LET H=C
2050 PRINT #Q,A$;TAB(2*W);H
2080 NEXT X
2081 LET Y=X-1
2082 PRINT #Q,TAB(2);"TOTAL CURRENT ASSETS";TAB(2*W+10);J
2090 RETURN
2099 :
3000 : FIXED ASSETS  150 -> 179
3001 :
3010 FOR X=Y TO T
3012 RECNO #10=X:GOSUB 9010::Get
3020 IF INT(A)>179 THEN 3081
3030 IF A<150 THEN 3080
3035 LET H=C
3050 LET K=K+H
3060 PRINT #Q,A$;TAB(2*W);H
3080 NEXT X
3081 LET Y=X-1
3082 PRINT #Q,TAB(2);"NET FIXED ASSETS";TAB(2*W+10);K
3090 RETURN
3099 :
4000 : OTHER ASSETS  180 -> 199
4001 :
4010 FOR X=Y TO T
4012 RECNO #10=X:GOSUB 9010::Get
4020 IF A >=199 THEN 4081
4030 IF A<180 THEN 4080
4040 LET H=C
4050 LET L=L+H
4060 PRINT #Q,A$;TAB(2*W);H
4080 NEXT X
4081 LET Y=X-1
4082 PRINT #Q,TAB(2);"TOTAL OTHER ASSETS";TAB(2*W+10);L
4090 RETURN
4099 :
5000 : CURRENT LIABILITIES  200 -> 219
5001 :
5010 FOR X=Y TO T
5012 RECNO #10=X:GOSUB 9010::Get
5020 IF A >219 THEN 5081
5030 IF A<200 THEN 5080
5040 LET H=C
5050 LET N=N+H
5060 PRINT #Q,A$;TAB(2*W);H
5080 NEXT X
5081 LET Y=X-1
5082 PRINT #Q,TAB(2);"TOTAL CURRENT LIABILITIES";TAB(2*W+10);N
5084 LET Z=Z-N
5090 RETURN
5099 :
6000 : LONG TERM LIABILITIES  220 ->299
6001 :
6010 FOR X=Y TO T
6012 RECNO #10=X:GOSUB 9010::Get
6020 IF A >299 THEN 6081
6030 IF A<220 THEN 6080
6040 LET H=C
6060 LET P=P+H
6070 PRINT #Q,A$;TAB(2*W);H
6080 NEXT X
6081 LET Y=X-1
```

More

Listing 4 continued.

```
6082 PRINT #Q,TAB(2);"TOTAL LONG TERM LIABILITIES";TAB(2*W+10);P
6084 LET Z=Z-P::NET PROFIT
6090 RETURN
6099 :
8000 : EQUITY  300 ->399
8001 :
8010 FOR X=Y+1 TO T
8012 RECNO #10=X:GOSUB 9010::Get
8020 IF A >398 THEN 8072
8040 LET H=C
8050 LET Q1=Q1+H
8052 LET Z=Z-H
8060 PRINT #Q,A$;TAB(2*W);H
8070 NEXT X
8072 IF Z>=0 P.#Q,"NET PROFIT";
8074 IF Z<0 P.#Q,"NET LOSS";
8076 PRINT #Q,TAB(2*W);Z
8082 LET Q1=Q1+Z
8084 PRINT #Q,TAB(2);"TOTAL STOCKHOLDERS' EQUITY";TAB(2*W+10);Q1
8090 RETURN
8099 :
9000 : PUT CHART.DAT
9001 :
9004 PUT #10,A,A$,B,C,D
9008 RETURN
9009 :
9010 : GET CHART.DAT
9011 :
9012 GET #10,A,A$,B,C,D
9018 RETURN
9019 :
9020 : GET   -> MASTER.DAT
9021 :
9022 GET #19,A,B,A$
9028 RETURN
9029 :
9030 : PUT   -> MASTER.DAT
9031 :
9032 PUT #19,A,B,A$
9038 RETURN
9039 :
9300 : BOOKS ARE OUT OF BALANCE
9301 :
9305 DIGITS= 2:RJUST=6:IF Q=1 THEN HOME
9310 PRINT #Q,"THERE IS AN ERROR IN THE BOOKS.":P.#Q
9320 PRINT #Q,"INCOME STATEMENT - PROFIT";TAB(2*W);"$ ";V7
9322 PRINT #Q,"BALANCE SHEET  -  PROFIT";TAB(2*W);"$ ";Z
9324 PRINT #Q
9326 PRINT #Q,TAB(W);"DIFFERENCE = ";TAB(2*W);"$ ";ABS(V7-Z)
9328 PRINT #Q
9330 PRINT #Q,"YOU MUST FIND THE ERROR BEFORE RUNNING THIS AGAIN."
9340 PRINT #Q,"LET'S GO FIND THE ERROR!"
9345 IF Q<>1 THEN SKIP #Q,15
9390 DIGITS= 0:RJUST=0
9395 CHAIN CHART.COR
```

Listing 5. TRANS.BAS program.

```
0001 : TRANS.BAS
0002 :
0003 : Embry  3/1/80
0004 :
0005 : Kb article
0006 :
0007 : Corporation
0009 :
0010 LINE= 0:W=25
0020 HOME
0024 PRINT "Posting of Transactions - Getting Data"
0030 GOSUB 9800::Pgm. Variables
0099 :
0100 : INITIALIZE
0101 :
0105 HOME
0110 PRINT TAB(W/2);"POSTING OF CHECKS & TRANSACTIONS"
0112 PRINT
0120 IF F1<>0 GOSUB 8900:GOTO 900
0140 PRINT "Transactions now on file = ";V0
0145 PRINT "Room for ";RSIZE #11-V0;"more."
0199 :
0200 : Menu
0201 :
0210 PRINT
0220 PRINT "1. Post an Intra-Chart Transaction"
0222 PRINT "2. Post a check"
0224 PRINT "3. Return to LEDGER control"
0226 PRINT
0250 INPUT "Make selection  ",S
0252 IF S>2 THEN 900
0260 ON S GOSUB 300,1000
0290 GOTO 100
```

More

Listing 5 continued.

```
0299 :
0300 : Input a Transaction
0301 :
0305 LET E1=0
0310 GOSUB 2000::get input
0312 IF F1<>0 THEN P.:GOTO 380
0320 GOSUB 4000::VERIFY
0322 IF Q$="y" THEN Q$="Y"
0324 IF Q$<>"Y" PRINT CHR$(7);"DATA NOT SAVED!":GOTO 370
0330 GOSUB 8100::Add
0332 IF F1<>0 GOSUB 8900:GOTO 370
0340 GOSUB 400::Pointers
0370 PRINT
0380 INPUT "Post another intra-chart transaction (Y/N) ",Q$
0382 IF Q$="y" THEN Q$="Y"
0384 IF Q$="Y" THEN 300
0390 RETURN
0399 :
0400 : Take care of pointers
0401 :
0420 RECNO #19=6:GOSUB 9030::GET
0430 LET B=B+1
0440 GOSUB 9040::PUT
0490 RETURN
0499 :
0899 :
0900 : Closing
0901 :
0990 CHAIN LEDGER.BAS
0999 :
1000 : Input a check
1001 :
1010 GOSUB 2000
1012 IF F1<>0 THEN 1080
1020 GOSUB 4000::VERIFY
1022 IF Q$="y" THEN Q$="Y"
1024 IF Q$<>"Y" PRINT CHR$(7);"DATA NOT SAVED!":GOTO 1080
1030 GOSUB 8100::ADD
1032 IF F1<>0 GOSUB 8900:GOTO 1080
1040 GOSUB 400::POINTERS
1080 PRINT
1082 INPUT "Want to post another check (Y/N) ",Q$
1084 IF Q$="y" THEN Q$="Y"
1086 IF Q$="Y" THEN 1000
1090 RETURN
1099 :
2000 : INPUT TRANSACTIONS & CHECKS
2001 :
2010 HOME
2011 IF S=2 P."Posting of Checks":P.
2012 IF S=2 THEN A1=100:GOTO 2022
2020 PRINT T$(1);TAB(W);:INPUT A1
2021 IF A1<100 P."ILLEGAL ACCOUNT.":GOTO 2020
2022 LET N=A1:GOSUB 3000
2024 IF F1<>0 GOSUB 8900:GOTO 2090
2026 LET I1=I::THE SLOT IN CHART.DAT FOR A1
2030 PRINT T$(2);TAB(W);:INPUT B1
2031 IF B1<100 P."ILLEGAL ACOUNT.":GOTO 2030
2032 LET N=B1:GOSUB 3000
2034 IF F1<>0 GOSUB 8900:GOTO 2090
2036 LET I2=I::THE SLOT IN CHART.DAT FOR B1
2040 PRINT T$(3);TAB(W);:INPUT E$
2050 PRINT T$(4);TAB(W);:GOSUB 2100::DATE
2060 PRINT T$(5);TAB(W);:INPUT D1
2070 IF S=2 P.T$(6);TAB(W);:INPUT E1
2090 RETURN
2099 :
2100 : INPUT DATE
2101 :
2105 PRINT
2110 INPUT "MONTH ",M
2112 IF M<1 THEN 2110
2114 IF M>12 THEN 2110
2116 LET M$=STR$(M)
2118 IF LEN(M$)<2 THEN M$="0"+STR$(M)
2120 INPUT "DAY ",D
2122 IF D<1 THEN 2120
2124 IF D>31 THEN 2120
2126 LET D$=STR$(D)
2128 IF LEN(D$)<2 THEN D$="0"+D$
2130 INPUT "YEAR ",Y
2132 IF Y<0 THEN 2130
2134 IF Y<100 THEN Y=Y+1900
2136 LET Y$=STR$(Y)
2138 IF LEN(Y$)<>4 THEN 2130
2140 LET T$=Y$+M$+D$
2150 LET C1=VAL(T$)
2190 RETURN
2199 :
3000 : BINARY SEARCH OF CHART.DAT
3001 :
3005 : N IS ACCT # FOR SEARCH
3006 :
3010 LET F1=0::ASSUME A SUCCESSFUL SEARCH
3012 IF N=399 THEN F1=3:GOTO 3090::NET PROFIT
3020 LET L=1:U=T
```

More →

Listing 5 continued.

```
3030 LET I=INT((L+U)/2)
3040 IF I=0 THEN F1=1:GOTO 3090
3050 IF L>U THEN F1=1:GOTO 3090
3052 RECNO #10=I:GOSUB 9000::GET
3060 IF N<A THEN U=I-1:GOTO 3030
3070 IF N>A THEN L=I+1:GOTO 3030
3090 RETURN
3099 :
4000 : VERIFY INPUT
4001 :
4010 HOME
4020 PRINT "VERIFY":P.
4030 FOR X=1 TO 6
4032 IF S=1 IF X=6 THEN 4072
4034 PRINT T$(X);TAB(W);
4040 IF X=1 P.A1
4045 IF X=2 P.B1
4055 IF X=3 P.E$
4060 IF X=4 P.C1
4062 IF X=5 DIGITS=2:P.D1:DIGITS=0
4064 IF X=6 IF S=2 P.E1
4070 NEXT X
4072 PRINT
4080 INPUT "DATA ENTRY O.K.  (Y/N) ",Q$
4090 RETURN
4999 :
5000 : UPDATE C.O.A.
5001 :
5010 LET X$=LEFT$(STR$(A1),1)
5012 LET Y$=LEFT$(STR$(B1),1)
5014 LET X=VAL(X$)
5016 LET Y=VAL(Y$)
5020 LET N=I1
5022 IF X<=3 GOSUB 9200:GOTO 5030
5024 GOSUB 9400
5030 LET N=I2:Z=D(Y,X)
5040 IF Y<=3 IF Z<0 GOSUB 9200:GOTO 5090
5050 IF Y<=3 GOSUB 9300:GOTO 5090
5060 IF Z<0 GOSUB 9400:GOTO 5090
5070 GOSUB 9300
5090 RETURN
5099 :
8100 : ADD
8101 :
8120 LET VO=VO+1
8122 IF VO>RSIZE #11 F1=4:GOTO 8190
8124 RECNO #11=VO:GOSUB 9020::Put
8130 GOSUB 5000::UPDATE ACCOUNTS
8140 IF VO=RSIZE #11 P."TRANSACTION FILE IS NOW FULL."
8190 RETURN
8199 :
8900 : Message Center
8901 :
8905 PRINT CHR$(7)
8910 IF F1=1 P."ACCT. # ";N;" DOES NOT EXIST!"
8930 IF F1=3 P."ACCT. # 399-NET PROFIT CAN'T BE CHANGED!"
8932 IF F1=4 P."TRANSACTION FILE IS NOW FULL. RE-ORGANIZE!"
8980 WAIT 5
8990 RETURN
8999 :
9000 : GET   ->  CHART.DAT
9001 :
9004 GET #10,A,A$,B,C,D
9008 RETURN
9009 :
9010 : PUT   ->  CHART.DAT
9011 :
9016 PUT #10,A,A$,B,C,D
9018 RETURN
9019 :
9020 : PUT   ->  TRANS.DAT
9021 :
9024 PUT #11,A1,B1,E$,C1,D1,E1
9028 RETURN
9029 :
9030 : GET   -> MASTER.DAT
9031 :
9032 GET #19,A,B,A$
9038 RETURN
9039 :
9040 : PUT  -> MASTER.DAT
9041 :
9042 PUT #19,A,B,A$
9048 RETURN
9049 :
9100 : ADD TO CHART.DAT
9101 :
9110 RECNO #10=N:GOSUB 9000
9120 LET C=C+D1
9140 GOSUB 9010
9190 RETURN
9199 :
9200 : SUB. FROM CHART.DAT
9201 :
9210 RECNO #10=N:GOSUB 9000
9220 LET C=C-D1
```

More

Reader Service—see page 226

Listing 5 continued.

```
9230 GOSUB 9010
9290 RETURN
9299 :
9300 : ADD TO CHART.DAT
9301 :
9310 RECNO #10=N:GOSUB 9000
9320 LET B=B+D1:C=C+D1
9330 GOSUB 9010
9390 RETURN
9399 :
9400 : SUB. FROM CHART.DAT
9401 :
9410 RECNO #10=N:GOSUB 9000
9420 LET B=B-D1:C=C-D1
9430 GOSUB 9010
9490 RETURN
9499 :
9800 : PGM VARIABLES
9801 :
9810 OPEN #19,0:MASTER.DAT
9814 DIM D(8,8),T$(6)
9816 RECNO #19=6:GOSUB 9030:V0=B::# IN TRANS.DAT
9820 OPEN #11,0:TRANS.DAT
9830 IF V0=RSIZE #11 THEN F1=4::TRANS.DAT IS FULL
9840 OPEN #10,0:CHART.DAT
9842 LET T=RSIZE #10
9850 FOR Y=1 TO 8:FOR X=1 TO 8:READ D(Y,X):NEXT X:NEXT Y
9860 FOR X=1 TO 6:READ T$(X):NEXT X
9890 RETURN
9899 :
9900 : Array to determine what to do
9901 :
9910 DATA 1,-1,-1,-1,1,1,-1,1
9912 DATA -1,1,1,1,-1,-1,1,-1
9914 DATA -1,1,1,1,-1,-1,1,-1
9916 DATA -1,1,1,1,-1,-1,1,-1
9918 DATA 1,-1,-1,-1,1,1,-1,1
9920 DATA 1,-1,-1,-1,1,1,-1,1
9922 DATA -1,1,1,1,-1,-1,1,-1
9924 DATA 1,-1,-1,-1,1,1,-1,1
9970 DATA THE 'FROM' ACCT. #:
9974 DATA THE 'TO' ACCT. #:
9976 DATA DESCRIPTION:
9978 DATA DATE:
9980 DATA AMOUNT:
9982 DATA CHECK #:
```


Listing 6. TRANS.PRT program.

```
0001 : TRANS.PRT
0002 :
0003 : Embry    3/1/80
0004 :
0005 : kb article
0006 :
0007 : Corporation
0008 :
0009 :
0010 HOME
0020 PRINT "Transaction print - Getting Data"
0030 GOSUB 9800::PGM VARIABLES
0040 IF V=0 P."NO TRANSACTIONS":GOTO 900
0099 :
0100 : INFORMATION
0101 :
0110 HOME
0120 PRINT "      This will print a list of all transactions"
0122 PRINT "since the data file was last cleared."
0130 PRINT
0140 INPUT "Is that what you want  (Y/N) ",Q$
0142 IF Q$="n" THEN Q$="N"
0144 IF Q$="N" THEN 900
0146 IF Q$="y" THEN Q$="Y"
0148 IF Q$<>"Y"THEN 140
0199 :
0200 : Menu
0201 :
0210 HOME
0212 PRINT "OPTIONS AVAILABLE"
0214 PRINT
0220 PRINT "1. Listing in order entered."
0226 PRINT "2. Return to Ledger control."
0228 PRINT
0230 INPUT "MAKE SELECTION ",S
0232 IF S>1 THEN 600
0240 INPUT "Which port for output  ",Q
0250 IF Q=1 THEN HOME
0260 GOSUB 300
0270 IF Q<>1 SKIP #Q,15
0290 GOTO 200
0299 :
```

More

Listing 6 continued.

```
0300 : LISTING OF TRANSACTIONS
0301 :
0310 LET L$="Transactions by ORDER ENTERED"
0312 GOSUB 1100::HEADER FOR TRANSACTIONS
0320 FOR X=1 TO V
0330 RECNO #10=X:GOSUB 9000::GET
0335 GOSUB 8900::CONVERT DATE
0340 GOSUB 2100::PRINT IT
0350 NEXT X
0388 LET H$="- ":GOSUB 1800
0390 RETURN
0399 :
0600 : RE-SET TRANS.DAT ?
0601 :
0610 PRINT
0620 INPUT "Do you wish to re-set the Transaction-Check Register file (Y/N)
0622 IF Q$="n" THEN Q$="N"
0630 IF Q$="N" THEN 900
0632 IF Q$="y" THEN Q$="Y"
0640 IF Q$<>"Y" THEN 620
0670 RECNO #19=6:GOSUB 9200
0680 LET A=0:B=0:GOSUB 9300
0699 :
0900 : Done
0901 :
0920 DLM=ON
0990 CHAIN LEDGER.BAS
0999 :
1100 : HEADER FOR INTRA-CHART TRANSFERS
1101 :
1110 LET H$="- ":GOSUB 1800
1112 PRINT #Q,TAB(35-LEN(N$)/2);N$:SKIP #Q,2
1116 PRINT #Q,TAB(35-LEN(L$)/2);L$
1118 SKIP #Q,2
1119 PRINT #Q,"TODAY'S DATE:";
1120 PRINT #Q,TAB(35-LEN(O$)/2);O$:SKIP #Q,2
1122 LET H$="= ":GOSUB 1800
1130 PRINT #Q,"   DATE";
1135 PRINT #Q, TAB(T1);"ACCOUNT #";
1140 PRINT #Q,TAB(T2+5);"DESCRIPTION";
1145 PRINT #Q, TAB(T3+3);"AMOUNT";TAB(T4+6);"CK. #"
1180 LET H$="= ":GOSUB 1800
1190 RETURN
1199 :
1800 : UNDERLINE
1801 :
1810 FOR X=1 TO T4/2+7
1820 PRINT #Q,H$;
1830 NEXT X
1840 PRINT #Q
1890 RETURN
1899 :
2100 : PRINT A TRANSACTION RECORD
2101 :
2103 IF X<10 P.#Q,TAB(2);
2105 LET X$=STR$(X)+". "+F$
2110 PRINT #Q,X$;
2112 RJUST= 7
2120 PRINT #Q,TAB(T1);E;
2130 LET K=E:GOSUB 8800
2140 PRINT #Q,TAB(T2);A$;
2148 DIGITS= 2:RJUST=7
2150 PRINT #Q,TAB(T3);H;
2151 DIGITS= 0
2152 IF I=0 P.#Q:GOTO 2156
2154 PRINT #Q,TAB(T4+5);I
2156 DIGITS= 0:RJUST=0
2158 IF S=3 THEN 2190
2159 RJUST= 7
2160 PRINT #Q,TAB(T1);F;
2170 LET K=F:GOSUB 8800
2175 PRINT #Q,TAB(T2);A$
2185 PRINT #Q,"PURPOSE: ";G$
2187 PRINT #Q
2190 RETURN
2199 :
8800 : BINARY SEARCH
8801 :
8810 LET L=1:U=T
8820 LET Y=INT((L+U)/2)
8822 SET RECNO #11=Y:GOSUB 9100::GET
8830 IF K=A THEN 8890
8840 IF K<A THEN U=Y-1:GOTO 8820
8850 LET L=Y+1
8860 GOTO 8820
8890 RETURN
8899 :
8900 : CONVERT DATE
8901 :
8910 LET F$=STR$(G)
8912 LET Y$=MID$(F$,3,2)
8920 LET F$=RIGHT$(F$,4)
8930 LET M$=LEFT$(F$,2)
8932 LET M=VAL(M$):M$=STR$(M)
8940 LET D$=RIGHT$(F$,2)
8942 LET D1=VAL(D$):D$=STR$(D1)
8950 LET F$=M$+"/"+D$+"/"+Y$
```

More →

Listing 6 continued.

```
8990 RETURN
8999 :
9000 : GET  ->  TRANS.DAT
9001 :
9010 GET #10,E,F,G$,G,H,I
9090 RETURN
9099 :
9100 : GET   -> CHART.DAT
9101 :
9110 GET #11,A,A$,B,C,D
9190 RETURN
9199 :
9200 : GET   ->  MASTER.DAT
9201 :
9210 GET #19,A,B,A$
9290 RETURN
9299 :
9300 : PUT   ->  MASTER.DAT
9301 :
9310 PUT #19,A,B,A$
9390 RETURN
9399 :
9800 : PGM VARIABLES
9801 :
9810 LET W=10:LINE=0:DLM=OFF
9812 OPEN #19,0:MASTER.DAT
9814 RECNO #19=6:GOSUB 9200:V=B::# IN TRANS.DAT
9817 IF V=0 THEN 9890::EMPTY
9818 RECNO #19=13:GOSUB 9200:N$=A$::NAME
9820 RECNO #19=14:GOSUB 9200:O$=A$::DATE
9822 PRINT "THERE ARE ";V;"TRANSACTIONS ON FILE."
9850 LET T1=15:T2=30:T3=55:T4=65
9880 OPEN #10,0:TRANS.DAT
9882 OPEN #11,0:CHART.DAT
9883 LET T=RSIZE #11
9890 RETURN
```

Listing 7. INIT.BAS program.

```
0001 : INIT.BAS
0002 :
0005 : Embry   3/1/80
0006 :
0007 : kb article
0009 :
0010 LINE= 0:W=43
0020 LET W1=7
0022 LET W2=20
0024 LET W3=55
0030 HOME
0032 PRINT "      This program will create the files necessary"
0034 PRINT "to run a minimum Accounting System.  The"
0036 PRINT "disk in drive #0 should contain only programs."
0038 PRINT "Three DATA files will be created.  If they now exist"
0040 PRINT "they will be DELETED!"
0042 PRINT
0044 INPUT "Do you want to continue  (Y/N)  ",Q$
0046 IF Q$<>"Y" IF Q$<>"N" PRINT CHR$(7):GOTO 44
0050 IF Q$="N" THEN END
0082 PRINT
0099 :
0100 : Count the Number of Accounts
0101 :
0110 READ N,N$
0120 IF N=0 PRINT"There are ";T;"accounts.":GOTO 150
0130 LET T=T+1
0140 GOTO 110
0150 IF T>255 PRINT"Maximum accounts is 255!  Re-orgaize.":END
0199 :
0200 : SET THE ARRAYS
0201 :
0205 RESTORE
0210 DIM C$(T),C(T,4)
0211 : C$(T) = This is the description of the Account
0212 : C(T,1) = This is the Account Number
0214 : C(T,2) = This is the Period or Current Balance
0216 : C(T,3) = This is the YTD Balance or Balance
0218 : C(T,4) = Reserved
0219 :
0220 FOR X=1 TO T
0230 READ N,N$
0240 LET C(X,1)=N:C$(X)=N$
0250 NEXT X
0299 :
0300 : Input the Balances of the accounts
0301 :
0310 INPUT "Do you want all account balances set to '0'  (Y/N)  ",Q$
0312 IF Q$="y" THEN Q$="Y"
0314 IF Q$="Y" THEN 500
0330 HOME
0340 PRINT "ACCT #      INPUT BALANCE AS REQUESTED."
```

More

Listing 7 continued.

```
0350 FOR X=1 TO T
0352 IF C(X,1)=399 THEN 399::Net Profit Exception
0380 PRINT C(X,1);"  ";C$(X);TAB(W);:INPUT"BALANCE  ",C(X,3)
0390 NEXT X
0399 :
0500 : Save the COA
0501 :
0510 LET Y$="0:CHART.DAT"
0512 IF FCHK Y$<>5 THEN FDEL Y$
0520 CREATE #10,0:CHART.DAT,T,58
0530 PRINT "*     Saving the COA    *"
0550 FOR X=1 TO T
0560 RECNO #10=X
0562 LET A=C(X,1):B=C(X,2):C=C(X,3):D=0
0570 PUT #10,A,C$(X),B,C,D
0580 NEXT X
0590 CLOSE #10
0599 :
0600 : Form the rest of the data files
0601 :
0610 LET Y$="0:MASTER.DAT"
0612 PRINT "Creating 'MASTER.DAT' file."
0614 IF FCHK Y$<>5 THEN FDEL Y$
0616 CREATE #10,MASTER.DAT,25,45
0618 CLOSE #10
0620 PRINT "Creating 'TRANS.DAT' file."
0622 LET Y$="0:TRANS.DAT"
0624 IF FCHK Y$<>5 THEN FDEL Y$
0626 CREATE #10,Y$,50,63
0628 CLOSE #10
0699 :
0700 : Input a few items
0701 :
0710 HOME
0712 LET A=0:B=0
0714 DLM=OFF
0718 OPEN #19,0:MASTER.DAT
0720 INPUT "Name of Corporation:  ",A$
0722 RECNO #19=13:PUT #19,A,B,A$
0730 INPUT "Street Address:       ",A$
0732 RECNO #19=22:PUT #19,A,B,A$
0740 INPUT "City, State  ZIP      ",A$
0742 RECNO #19=23:PUT #19,A,B,A$
0799 :
0800 : Save the Months
0801 :
0810 READ Y$::step over the last two '0'
0820 FOR X=1 TO 12
0822 READ Y$
0824 RECNO #19=X
0826 PUT #19,A,B,Y$
0830 NEXT X
0890 CLOSE #19
0899 :
0900 : MESSAGE
0901 :
0910 HOME
0920 PRINT "     This completes the initialization of the Chart"
0922 PRINT "of Accounts.  You may re-run this program anytime by"
0924 PRINT "loading in BASIC and typing 'CHAIN INIT.BAS'."
0970 DLM=ON
0990 CHAIN BEGIN
0999 :
1000 : Current Assets
1001 :
1005 : ACCTS. 100 -> 149
1009 :
1010 DATA 100,CHECKING
1020 DATA 103,PETTY CASH
1036 DATA 106,PREPAID INSURANCE
1038 DATA 107,PREPAID INTEREST
1039 DATA 108,INSURANCE - CASH VALUE
1070 DATA 110,A/R - TRADE
1072 DATA 111,A/R - EMPLOYEES
1080 DATA 120,INVENTORY
1099 :
1999 :
2000 : Fixed Assets
2001 :
2002 : Accounts    150  -> 179
2009 :
2012 DATA 150,LAND
2014 DATA 160,DEPRECIABLE ASSETS
2016 DATA 161, LESS DEPRECIATION CLAIMED
2999 :
3000 : Other Assets
3001 :
3005 : ACCTS. 180 -> 199
3009 :
3010 DATA 180,DEFERRED INTEREST EXPENSE
3020 DATA 185,ORGANIZATIONAL EXPENSE
3999 :
4000 : Current Liabilities
4001 :
4005 : ACCTS. 200 -> 219
4009 :
4010 DATA 200,ACCOUNTS PAYABLE
```

More

Listing 7 continued.

```
4030 DATA 205,FICA WITHHELD
4040 DATA 206,FED. TAX WITHHELD
4050 DATA 207,STATE TAX WITHHELD
4055 DATA 208,STATE - UNEMP TAX
4057 DATA 209,FED. UNEMP TAX
4090 DATA 217,INTEREST PAYABLE
4099 :
4999 :
5000 : LONG TERM LIABILITIES
5001 :
5005 : ACCTS. 220 -> 299
5009 :
5010 DATA 220,NOTE #1 PAYABLE
5020 DATA 222,NOTE #2 PAYABLE
5999 :
6000 : STOCKHOLDERS' EQUITY
6001 :
6005 : ACCTS. 300 -> 399
6009 :
6010 DATA 300,CAPITAL
6030 DATA 310,RETAINED EARNINGS
6090 DATA 399,NET PROFIT
6999 :
7000 : SALES &/OR INCOME
7001 :
7005 : ACCTS. 400 -> 499
7009 :
7010 DATA 400,CASH - SALES
7020 DATA 401,SALES - ON ACCOUNT
7499 :
7500 : INVENTORY
7501 :
7506 :
7510 DATA 500,COST OF MDSE. SOLD
7999 :
8000 : OPERATING EXPENSES
8001 :
8005 : ACCTS. 600 -> 699
8009 :
8010 DATA 600,ADVERTISING & SALES PROM.
8012 DATA 602,ACCOUNTING & AUDIT FEES
8014 DATA 604,BANK SERVICE CHARGE
8016 DATA 606,DATA PROCESSING
8020 DATA 610 DEPRECIATION
8024 DATA 614,DONATION
8026 DATA 616,DUES & SUBSCRIPTION
8028 DATA 617,FREIGHT
8032 DATA 620,INSURANCE
8036 DATA 624,INTEREST
8042 DATA 630,LEGAL
8046 DATA 634,MAINTENANCE
8049 DATA 637,INVENTORY SHORTAGE
8050 DATA 638,OFFICE SUPPLIES
8052 DATA 640,PAYROLL
8060 DATA 648,PAYROLL TAXES
8070 DATA 658,POSTAGE
8072 DATA 660,RENT
8078 DATA 666,TELEPHONE & TELEGRAPH
8080 DATA 668,TRAVEL & ENTERTAINMENT
8088 DATA 676,UTILITIES
8099 :
8499 :
8500 : OTHER INCOME
8501 :
8505 : ACCTS 700 -> 799
8509 :
8510 DATA 700,INTEREST INCOME
8520 DATA 710,RENTAL INCOME
8530 DATA 720,CASH OVER OR SHORT
8540 DATA 730,SALE OF ASSETS
8999 :
9000 : OTHER EXPENSES
9001 :
9005 : ACCTS. 800 -> 899
9009 :
9010 DATA 805,BAD DEBTS
9020 DATA 810,PENALITIES
9030 DATA 820,COST OF ASSETS SOLD
9040 DATA 830,MISCELLANEOUS EXPENSES
9090 DATA 0,0
9099 :
9100 : Months
9101 :
9110 DATA January
9112 DATA February
9114 DATA March
9116 DATA April
9118 DATA May
9120 DATA June
9122 DATA July
9124 DATA August
9126 DATA September
9128 DATA October
9130 DATA November
9132 DATA December
```

More

Listing 8. CHART.COR program.

```
0001 : CHART.COR
0002 :
0003 : Embry  3/1/80
0004 :
0005 : Kb article
0006 :
0007 : Corporation
0008 :
0010 HOME
0012 LINE= 0:W=30
0030 PRINT "Getting Data."
0040 DLM=OFF
0090 GOSUB 1000::Get Ready
0099 :
0100 : Introduction
0101 :
0105 HOME
0110 PRINT "     This program will permit you to obtain a"
0112 PRINT "listing of the current status of your Chart-Of-Accounts."
0114 PRINT "You will also be given the option of making any"
0116 PRINT "corrects you think necessary."
0130 PRINT
0140 PRINT "IF YOU DON'T KNOW WHAT YOU ARE DOING CALL YOU SUPERVISOR."
0150 PRINT
0160 INPUT "DO YOU KNOW WHAT TO DO  (Y/N)  ",Q$
0162 IF Q$="n" THEN Q$="N"
0164 IF Q$="N" THEN 900
0166 IF Q$="y" THEN Q$="Y"
0168 IF Q$<>"Y" P."YOU ENTERED ";Q$;". DO IT RIGHT.":GOTO 150
0170 RECNO #19=14:GOSUB 9020
0172 LET D$=A$::DATE
0190 RECNO #19=13:GOSUB 9020:W$=A$::NAME
0199 :
0200 : Selection
0201 :
0205 HOME
0210 PRINT TAB(W/2);"CHART OF ACCOUNT CORRECTION PROGRAM"
0211 PRINT
0212 PRINT "1. List Chart-of-Accounts"
0218 PRINT "2. Make corrections to Chart-of-Accounts"
0219 PRINT "3. Return to Ledger control"
0220 PRINT
0222 INPUT "MAKE SELECTION  ",S
0224 IF S<1 THEN 200
0226 IF S>2 THEN 900
0230 IF S=1 GOSUB 8900::PORT ?
0280 ON S GOSUB 8000,8300
0290 GOTO 200
0299 :
0900 : Done
0901 :
0910 DLM=ON
0990 CHAIN LEDGER.BAS
0999 :
1000 : Get Ready
1001 :
1010 OPEN #19,0:MASTER.DAT
1020 OPEN #11,0:CHART.DAT
1022 LET T=RSIZE #11
1030 PRINT "You have ";T;"accounts."
1040 WAIT 5
1090 RETURN
1099 :
2000 : Change COA
2001 :
2005 HOME
2010 PRINT "     <  ASSETS = LIABILITIES + EQUITY >"
2011 PRINT
2012 PRINT "ITS YOUR JOB TO KEEP IT BALANCED!!!"
2014 PRINT
2030 GOSUB 7000::do Binary Search
2040 IF F1<>0 THEN 2070
2050 GOSUB 3000::VIDEO AND CHANGE
2052 GOSUB 9000::PUT
2060 GOTO 2090
2070 PRINT CHR$(7)
2080 PRINT "ACCT. # ";N;" DOES NOT EXIST."
2090 RETURN
2099 :
3000 : Video  &  Change
3001 :
3010 HOME
3020 PRINT "1.  ACCOUNT #";TAB(W);A
3022 PRINT "2.  DESCRIPTION";TAB(W);A$
3023 DIGITS= 2
3024 IF A<400 THEN PRINT:GOTO 3026
3025 PRINT "3.  CURRENT BALANCE";TAB(W);B
3026 PRINT "4.  ACCOUNT TOTAL";TAB(W);C
3027 DIGITS= 0
3028 PRINT
3030 INPUT "IF O.K. ENTER '0' ELSE NUMBER TO CHANGE ",Y
3032 IF Y=1 P."CAN'T CHANGE ACCOUNT #.":WAIT 5:GOTO 3090
3035 IF Y<=.5 THEN 3090
```

More

Listing 8 continued.

```
3040 PRINT "ENTER CORRECTION  ";
3044 IF Y=2 INPUT A$
3046 IF Y=3 INPUT B
3048 IF Y=4 INPUT C
3050 IF Y>4 THEN 3030
3060 GOTO 3000
3090 RETURN
3099 :
5000 : Print COA
5001 :
5010 FOR X=1 TO T
5011 RECNO #11=X:GOSUB 9010
5012 PRINT #Q,X;
5015 PRINT #Q,TAB(W1);A;
5020 PRINT #Q,TAB(W2);A$;
5030 GOSUB 9200::$ TO PRINT
5032 IF A<400 THEN 5044
5040 PRINT #Q,TAB(W3);B;
5044 PRINT #Q,TAB(W4);C
5050 GOSUB 9300::no $ to Print
5055 IF IMOD(X,5)=0 THEN PRINT #Q
5060 NEXT X
5090 RETURN
5099 :
7000 : Binary Search
7001 :
7010 LET L=1:U=T
7020 LET I=INT((L+U)/2)
7022 IF I=0 THEN 7080
7024 IF L>U THEN 7080
7030 SET RECNO #11=I:GOSUB 9010
7040 IF N=A THEN F1=0:GOTO 7090::Found
7050 IF N<A THEN U=I-1:GOTO 7020
7060 IF N>A THEN L=I+1:GOTO 7020
7080 LET F1=1::Error Flag
7090 RETURN
7099 :
8000 : List COA
8001 :
8005 IF Q=1 THEN HOME
8007 IF Q<>1 THEN PRINT #Q
8010 GOSUB 9600::Print Header
8020 GOSUB 9400::Label Columns
8030 GOSUB 5000::Get and Print information
8040 LET L$="==":GOSUB 9500
8080 IF Q<>1 THEN SKIP #Q,15
8090 RETURN
8099 :
8300 : Make Corrections
8301 :
8310 PRINT
8320 INPUT "Which Account # to Modify  ",N
8330 GOSUB 2000::Search & Change
8340 PRINT
8380 INPUT "Want to Modify another account  (Y/N)  ",Q$
8382 IF Q$="Y" THEN 8300
8384 IF Q$="y" THEN 8300
8390 RETURN
8399 :
8900 : Port-of-Print
8901 :
8910 LET W1=8:W2=16
8920 INPUT "Which port for output  ",Q
8930 IF Q=1 THEN W3=49:W4=59
8940 IF Q<>1 THEN W3=59:W4=69
8990 RETURN
8999 :
9000 : PUT   -> CHART.DAT
9001 :
9005 PUT #11,A,A$,B,C,D
9008 RETURN
9009 :
9010 : GET    ->   CHART.DAT
9011 :
9012 GET #11,A,A$,B,C,D
9018 RETURN
9019 :
9020 : GET   ->  MASTER.DAT
9021 :
9022 GET #19,A,B,A$
9028 RETURN
9029 :
9200 : $ TO PRINT
9201 :
9210 DIGITS= 2
9230 RJUST= 6
9290 RETURN
9299 :
9300 : NOT $ TO PRINT
9301 :
9310 DIGITS= 0
9320 RJUST= 0
9390 RETURN
9399 :
9400 : PRINT HEADER FOR C.O.A.
9401 :
```

More

Listing 8 continued.

```
9410 PRINT #Q,"ACCT. #";
9420 PRINT #Q,TAB(W2-2);"DESCRIPTION";
9430 PRINT #Q,TAB(W3);"CURRENT";
9440 PRINT #Q,TAB(W4+3);"BALANCE"
9450 LET L$="= ":GOSUB 9500
9490 RETURN
9499 :
9500 : Print L$
9501 :
9510 FOR X=1 TO (W4+10)/2
9520 PRINT #Q,L$;
9530 NEXT X
9540 PRINT #Q
9590 RETURN
9599 :
9600 : Header for report
9601 :
9610 PRINT #Q,TAB(35-LEN(W$)/2);W$
9620 SKIP #Q,2
9630 PRINT #Q,TAB(35-LEN(D$)/2);D$
9640 SKIP #Q,2
9650 LET L$="**":GOSUB 9500
9690 RETURN
```

Listing 9. RESET.COA program.

```
0001 : RESET.COA
0002 :
0003 : Embry  3/1/80
0004 :
0005 : Kb article
0006 :
0007 : Corporation - Service
0008 :
0010 HOME
0099 :
0100 : Instructions
0101 :
0110 PRINT "THIS WILL RESET TO ZERO = 0 ALL(!!!!) DOLLARS"
0112 PRINT "IN THE CHART-OF-ACCOUNTS!"
0114 PRINT
0120 INPUT "IS THIS WHAT YOU WANT  (Y/N) ",Q$
0122 IF Q$="N" P."I THOUGHT NOT.":GOTO 990
0124 IF Q$<>"Y" THEN 120
0130 INPUT "ENTER PASSWORD AND IT SHALL BE DONE ",Q$
0132 IF Q$<>"PASSWORD" P."WRONG PASSWORD - BYE!":END
0199 :
0200 : DO IT
0201 :
0210 OPEN #10,1:CHART.DAT
0212 PRINT "# OF ACCOUNTS = ";RSIZE #10
0220 FOR X=1 TO RSIZE #10
0222 IF IMOD(X,10)=0 P.X;
0228 SET RECNO #10=X
0230 GET #10,A,A$,B,C,D
0240 LET B=0:C=0
0250 PUT #10,A,A$,B,C,D
0260 NEXT X
0990 CHAIN BEGIN
```

Listing 10. MASTER.CHG program.

```
0001 : MASTER.CHG
0002 :
0003 : Embry  3/1/80
0004 :
0005 : Kb article
0006 :
0008 :
0020 LINE= 0
0030 GOSUB 9800::Pgm. Variables
0099 :
0100 : INTRO.
0101 :
0110 HOME
0120 PRINT "This will allow you to check & change the"
0122 PRINT "data in a file named 'MASTER.DAT'."
0130 PRINT
0140 INPUT "WANT A PRINT-OUT  (Y/N) ",Q$
0142 IF Q$="y" THEN Q$="Y"
0144 IF Q$="Y" GOSUB 7500::Print out wanted
0199 :
0200 : SELECT
0201 :
0205 HOME
0207 PRINT "ENTER A '0' TO TERMINATE."
```

More

Listing 10 continued.

```
0210 INPUT "WHICH RECORD # TO CHECK  ",R
0212 IF R=0 THEN 900
0214 IF R>RSIZE #19 THEN 900
0220 RECNO #19=R
0230 GET #19,A,B,A$
0240 PRINT
0250 PRINT A$(R,1);TAB(W);A
0252 PRINT A$(R,2);TAB(W);B
0256 PRINT A$(R,3);TAB(W);A$
0260 PRINT
0262 INPUT "O.K.  (Y/N) ",Q$
0264 IF Q$="Y" THEN 290
0266 IF Q$="y" then 290
0270 PRINT "NEW VALUES"
0272 INPUT "A = ",A
0274 INPUT "B = ",B
0276 INPUT "A$= ",A$
0280 PUT #19,A,B,A$
0290 INPUT "ANOTHER (Y/N) ",Q$
0310 IF Q$="N" THEN 900
0312 IF Q$="n" then 900
0330 GOTO 200
0399 :
0900 : DONE
0901 :
0910 INPUT "Want a print out  (Y/N) ",Q$
0912 IF Q$="Y" GOSUB 7500
0914 IF Q$="y" GOSUB 7500
0970 CLOSE #19
0980 DLM=ON
0990 CHAIN LEDGER.BAS
0999 :
1000 : DATA
1001 :
1010 DATA RESERVED,RESERVED,MONTH
1020 DATA TODAY'S DATE FORMATTED,JULIAN DAY,MONTH
1030 DATA RESERVED,RESERVED,MONTH
1040 DATA RESERVED,RESERVED,MONTH
1050 DATA RESERVED,RESERVED,MONTH
1060 DATA RESERVED,# IN TRANS.DAT,MONTH
1070 DATA PROFIT - P&L,END OF MONTH FLAG,MONTH
1080 DATA RESERVED,RESERVED,MONTH
1090 DATA RESERVED,RESERVED,MONTH
1100 DATA RESERVED,SPARE,MONTH
1110 DATA RESERVED,RESERVED,MONTH
1120 DATA RESERVED,RESERVED,MONTH
1130 DATA SPARE,SPARE,NAME
1140 DATA SPARE,SPARE,TODAY-READABLE
1150 DATA SPARE,SPARE,SPARE
1160 DATA SPARE,SPARE,SPARE
1170 DATA SPARE,SPARE,SPARE
1180 DATA SPARE,SPARE,SPARE
1190 DATA SPARE,SPARE,SPARE
1200 DATA SPARE,SPARE,SPARE
1210 DATA SPARE,SPARE,RESERVED
1220 DATA SPARE,SPARE,ADDRESS
1230 DATA SPARE,SPARE,CITY-STATE-ZIP
1240 DATA SPARE,SPARE,RESERVED
1250 DATA SPARE,SPARE,RESERVED
1299 :
7500 : PRINT OUT
7501 :
7510 INPUT "Which port for output  ",Q
7512 IF Q<>3 THEN HOME
7513 : Sets DEC LA-34 for 8 lines/inch and 10 characters/inch
7514 IF Q<>1 PRINT #Q,ESC(91);CHR$(51);CHR$(119)
7518 PRINT #Q
7520 PRINT #Q,TAB(W-LEN(N$)/2);N$
7522 PRINT #Q,TAB(W-LEN(D$)/2);D$
7524 SKIP #Q,2
7530 PRINT #Q,"REC.#";TAB(W);"Description & Value"
7540 FOR X=1 TO RSIZE #19
7550 RECNO #19=X
7552 GET #19,A,B,A$
7568 PRINT #Q,X;" ";
7570 PRINT #Q,A$(X,1);"=";A;TAB(W1);A$(X,2);"=";B;TAB(W2);A$(X,3);"=";A$
7572 IF IMOD(X,5)=0 PRINT #Q
7580 NEXT X
7590 RETURN
7599 :
9800 : Pgm. variables
9801 :
9805 DLM=ON ::set delimiter to ','
9810 LET W=38
9812 LET W1=37:W2=65
9830 DIM A$(25,3)
9832 FOR X=1 TO 25
9834 READ A$(X,1),A$(X,2),A$(X,3)
9836 NEXT X
9880 DLM=OFF
9882 OPEN #19,O:MASTER.DAT
9884 RECNO #19=14
9886 GET #19,A,B,D$::date
9888 LET N$="MASTER.DAT File Contents"
9890 RETURN
```

✓ Reader Service—see page 226

This secretary has a way with words.

Word Processor for the North Star Disk System

By G. L. Haller

I recently replaced my Sol computer with a new Horizon II that uses the standard serial terminal. But I realized afterwards that I would not be able to use my favorite word processor, which requires memory mapping.

At first, I considered adding a memory-mapped interface board to the Horizon. But that would have meant converting my terminal to accept video input and would almost have amounted to a double system.

I then decided to see what could be done with programs that did not require memory mapping.

I studied the descriptions of several, and of those tested, Maryelln by Gary Young best suited my needs. At about the same time, Young announced a much-improved version, which he calls Secretary. Since receiving and testing it, I am satisfied that I need not worry about my lack of a memory-mapped system.

Briefly, Secretary uses a line-oriented program and can include the formatting commands in the program.

Configuring

When you first run the disk, it will ask if it is to be reconfigured. After it is, the disk will not ask again, but may be reconfigured at any time by the command CONFIGURE. The Secretary program will then be set to the computer memory size available, the type of character to be used for back space, single or double density, a stop at the end of each page, lines per screen, screen line length and the line wrap-around.

This latter feature, when in the AUTO line-numbering mode, will automatically go to the next line when the line length of the terminal is used. In any case, the bell will sound when you are within seven characters of the end of the line. The Secretary program may then be saved on disk for future use.

Secretary is written in assembly language for the North Star Disk System. It starts at 2D00 and may be configured for single or double density.

Pre-Text Formatting

The pre-text formatting commands are:

- JUSTIFY, which can be turned on or off.
- LINE XX, setting the length of the line.
- MARGIN XX,XXX, which sets the right and left margins.
- OFFSET XX, which sets the left margin only.
- PAGE XX,XX,XX, which defines a page, the maximum lines, the number of printed lines and the line number of the first line of the text.
- PNUM XX, starting the numbering of the pages.
- SPACE XX, setting the spaces between lines.
- TABSET X1, X2, and so on, which sets up to ten tabs.
- TITLE TEXT1, which sets the title to be printed at the top of each page.

Many of the above commands, while principally used pre-text, may also be used combined with the text.

In-Text Formatting

1. An in-text command allows indenting with either right- or left-hand margins or both.
2. A left arrow centers the text with respect to the line length.
3. Tabs that follow the TABSET command can be used.
4. Data can be merged from a BASIC file. This is done in sequence by the use of an up arrow in the text. This is especially useful in form letters with different addresses or comments in the text.
5. Underlining may be accomplished by using the ampersand before and after the text to be underlined.
6. Brackets around any portion of text will evoke the JUSTIFY command or will line-fill if justification is not desired.

Editing

Editing is done with the North Star editing conventions and a few extra ones. The usual North Star commands used are:

- Control-G, copy to the end of the line.
- Control-D, copy to a specific character.
- Control-Z, delete a character.
- Control-A, copies one character.
- Control-Q, back space one character.
- Control-Y, toggle the insert mode.
- Control-N or control-P, reedit the old line.
- Control-C, exit the command with the line unaltered.

Other commands not in North Star include:

- FIND TEXT1, which locates all occurrences of TEXT1.
- CHNG, which changes selected text and waits for approval or non-approval.
- CHAL, which changes all selected text without approval.
- COPY, which copies text in newly selected lines but leaves the old lines.
- MOVE, which copies text in newly selected lines and deletes old lines.
- LIST command, which will list to any output device. See UNIT command.
- RENUMBER, which renumbers lines.
- OPEN, which opens up line num-

Dr. G. L. Haller, 1500 Galleon Drive, Naples, FL 33940.

bers in selected line sequences for insertion of more lines of text.
- SCRATCH, which deletes all lines.
- APPEND, which may be used to bring two programs together. Then, by the use of the OPEN and MOVE commands, the text may be placed as desired.
- DELETE, will delete a line or a series of lines on command.

Saving Programs to Disk

Again, the North Star conventions are used with respect to saving and loading from disk files. By using the FREE command, not only the length of the program is given, but also the number of blocks, so that the NSAVE command may be used efficiently. More than one disk drive can be used in the conventional way. A DIRECTORY command will output a directory of the files, and in the DOS operation, the LI command will also list the text files.

Special Commands

Quite a few special commands are helpful in the operation of any program. They include:
- HELP, which will list the first four letters of all valid commands. The first four letters are all that are needed for a command. Either lowercase or uppercase letters may be used. This listing is also done automatically after an invalid command.
- NULL XX, which will generate XX nulls after each C/R.
- QUIT, which will leave the program and return to DOS. From DOS, a JP 2D00 will go back to Secretary; a JP 2D04 will go back to Secretary and retain the text program in memory.
- CALL, which allows a call to a machine language program in the computer memory. A sample of this is in the Secretary program.
- FREE, which will print the number of bytes used, the number still available, the last line number and the blocks used.
- A NULL command to insert a specified number of nulls following each carriage return.
- RECOVER, which will attempt to recover programs destroyed by operator error.
- COMMENT, which can be embedded in the program to have nonprinting comments in the listing.
- MERGING, which may be done from a separate BASIC file for inserting material such as sequential addresses or comments into the program. A BASIC program is provided to produce the BASIC files for merging.
- A back slash to terminate a page. If a number follows the back slash, the program will look to see if that many lines can still be printed on that page; if so, it will disregard the back slash.

Messages

Forty-four messages will aid you in using the Secretary program. These messages will indicate that your commands have been accepted, or that the file in disk is too small to accept a save, or a file in disk is too large for your memory, or many others.

Listing and Printing

The UNIT command will cause the program to output on the desired output device. Unit 1 will output on the #1 output device, and Unit 0 will output on the #0 device (usually the video terminal). When in the Unit 0 mode, the output will stop at screen full, and wait for a Return to continue for both listing and printing. On any other Unit number the output is continuous, but printing follows the directions of the PAGE commands on the video screen so that you can observe what the final format will look like. Be sure to return to Unit 0 for a video listing after a hard-copy printing.

Miscellaneous Comments

I have set up several formats of pretext short programs so that when I know what letter form I will use, I simply load that format and add the text.

There is only one minor irritation in running the program, and it is in the editing mode. When searching for a character with the control-D command, if you are looking for the second occurrence of the same character you must use control-A to get the cursor off the first find.

Conclusion

The documentation that comes with the program is complete and includes a 30-page book of instructions. Two text files that illustrate the entire range of the features of Secretary are included. A thorough instruction lesson is provided by listing and printing these two programs.

The program is well worth $85. It may be bought from the author, Gary Young, PO Box 3218, North Hollywood, CA 91609. I purchased mine from American Square Computers, Jamestown, NC 27282.■

✓ Reader Service—see page 226

In all my dealings with software houses the one company which has always impressed me with its high quality service has been

Instant Software™

Scott Adams
10/2/80

SCOTT ADAMS,
President of
Adventure International
and author of the
"Adventure" Series.

Instant Software™

PETERBOROUGH, NEW HAMPSHIRE 03458 603-9247296

✓ Reader Service—see page 226

KILOBAUD KLASSROOM NO. 24

Connecting to the Outside World

Now that you've learned something of assembly- and machine-language programming, you're ready to tackle some complete programs.

Peter A. Stark
PO Box 209
Mt. Kisco, NY 10549

Last month we covered assembly language for our Kilobaud Klassroom Komputer and gave some examples of simple programming. But no program is complete without some input and output. Since the primary I/O device for our computer is the 6820/6821 PIA, let's examine that next.

The PIA

The PIA is Motorola's parallel interface adapter IC, a parallel I/O device specifically designed to work with processors in the 6800 family. It is available under the numbers 6820 or 6821; it is also identical with the 6520, which is designed to go with the 6500-series processors.

As shown in Fig. 1, the PIA has a group of pins which connects to the 6800 (shown on the left) and another group of pins which connects to external devices (shown on the right). In most applications we are concerned with the latter group.

The 20 I/O pins shown at the right are divided into ten for port A and ten for port B. Except for some slight electrical differences, the two ports are identical in most applications.

Each port contains eight data lines and two handshaking lines. The data lines can be used for both input or output (though not at the same time) in any combination. For example, of the eight lines on one port, three could be used for input, while the other five are outputs.

In addition, two more lines on each port are the handshaking lines. One of these must be used for input, while the other can be either an input or an output. While they can be used for general-purpose I/O as well, thei programming is a bit more complex than the eight data lines and so they are usually reserve just for handshaking in paralle systems. In simple application (such as controlling a heating system or burglar alarm) the would probably not be used unless needed.

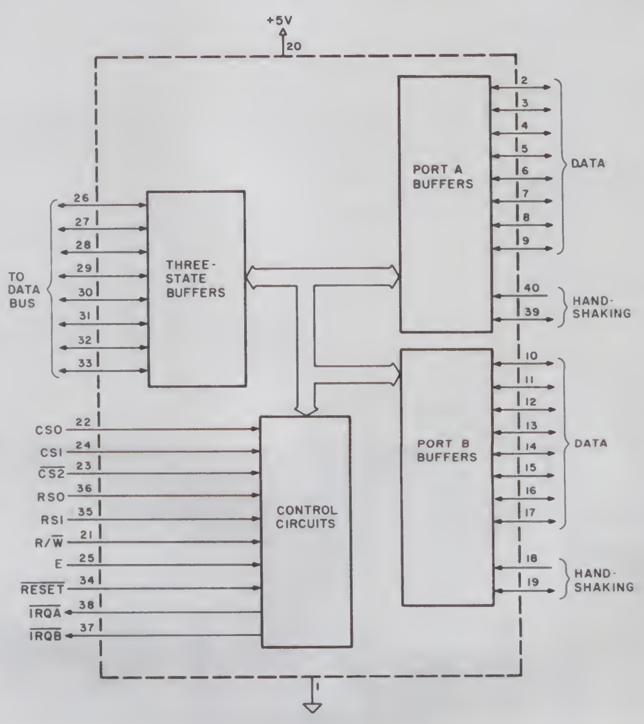


Fig. 1. PIA block diagram.

All of these lines are said t be "TTL compatible"; that is, th signals on them are 0 volts an +5 volts, such as those of TT logic. Though they can connec directly to TTL devices, they d not have the current-handlin capabilities of TTL devices Moreover, the PIA is an MOS de vice and can thus be damage by static electricity. Hence some kind of driver amplifier o isolation device will be used be tween PIA pins and the outsid world.

PIA Programming

In our system, the PIA is con nected to the address bus of th computer in two ways. First, it $\overline{CS2}$ chip select signal is drive from an output on the 74LS13 address decoder. There is provi sion for two PIAs on the board but let's consider only the first

```
START CLR    $A001  Clear port A control
      LDA A  #$07
      STA A  $A000  Store 00000111 into direction
      LDA A  #$04
      STA A  $A001  Store 00000100 into control
```

Listing 1.

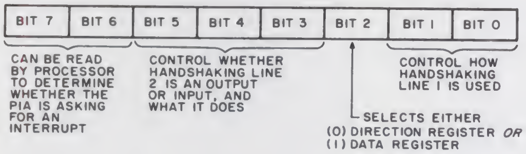


Fig. 2. The PIA control register.

This one is driven from the 74LS138 output which goes low for addresses in the range from A000 through BFFF. Thus, the PIA occupies this entire range of addresses, and any instruction which references these addresses will, in some way, activate the PIA.

In addition, however, the RS0 line shown in Fig. 1 connects to bit A0 of the address bus, and RS1 connects to A1. The status of these two address lines is also received by the PIA, and affects it in different ways. These two bits determine which part of the PIA is addressed.

Since these two address lines can take on any one of four values (00, 01, 10 or 11), the PIA responds to these as four different addresses. For example, the hex address A000 ends with the bits 00, whereas hex address A003 ends with the bits 11. Thus, the addresses A000 through A003 access the PIA in four different ways.

You'll note that address A004 again ends with the bits 00, just as A000 did; so does A008, A00C, A010 and so on. As it turns out, there are 2K addresses in the range of A000–BFFF that end with 00, another 2K that end with 01, 2K that end with 10, and 2K that end with 11. There are many different addresses we could use to refer to the PIA, but just to keep it simple, from now on we will use only the addresses A000–A003 and ignore the rest (even though the PIA does not).

Of these addresses, A000 and A001 apply only to port A, while A002 and A003 apply only to port B. The four addresses are used as follows:

A000—Port A data and direction registers
A001—Port A control register
A002—Port B data and direction registers
A003—Port B control register

Each port has two addresses, and, except for one small difference (having to do with the handshaking lines), the two ports are programmed in identical ways.

First, you will note that addresses A001 and A003 are used for just one register each, while A000 and A002 service two registers each. Though this seems like an unnecessary complication, in actual use it is not so inconvenient. When power is first applied, these addresses are used for the direction registers, while after some initialization they apply only to the data registers.

The direction registers determine which bit of which port is an output pin and which is an input. As mentioned earlier, any line on either port can be either an input or an output, though not at the same time. The decision as to which bit is which is made by the direction register. If a given bit of the direction register is 0, the corresponding bit of the port is an input. If the bit is a 1, then that bit of the port is an output. (The easiest way to remember this is by writing 0 = I—0 = Input and Output = 1. It's just the opposite of what you would expect.)

For example, the following instructions

```
LDA A #$07    Load 07 into A acc
STA A $A000   Store into port A Dir. Reg
```

place a binary 00000111 (hex 07) into the direction register. This makes the first five bits of the port into inputs, and the last three bits into outputs.

The data registers are used for actual communication with each port. To output from the port, the program simply stores data into the data register; to input from the port, it simply loads data from the register.

The control register is used for two-way communication between the processor and the PIA. Each of the bits in this register has a specific use, and by using the control register, the processor can manipulate the handshaking lines, control the generation of interrupts by the PIA and manipulate the PIA in other ways, as well as sense what the PIA is doing.

Fig. 2 shows the meaning of the eight bits in the control register. Five bits are used by the processor to control the two handshaking lines of each port, while two other bits can be read by the processor to determine whether this PIA port is asking for an interrupt.

Of special interest to us right now is bit 2. As you remember, the PIA data and direction registers share one address; bit 2 of the control register determines which of these two is actually used by that address. If this bit is 0, then the direction register is selected; if this bit is 1, then the data register is selected.

When the computer is first started, the same RESET signal which resets and initializes the 6802 processor also resets all the registers of the PIA to 0. At the beginning, bit 2 of the control register is automatically 0, and so addresses A000 and A002 apply to the direction registers. One of the first steps of a program would therefore initialize these registers to tell the PIA how it is going to be used.

Once this initialization is done, the program would normally store 00000100, or hexadecimal 04, into the control register. This sets bit 2 so that from now on the processor accesses the data register instead of the direction register.

Reader Service—see page 226

Listing 2. ACTESTER—the acoustic coupler test program.

```
                    NAM   LISTING   2

                ************************************
                * ACOUSTIC COUPLER TESTER PROGRAM *
                * FOR THE KILOBAUD KLASSROOM       *
                * SINGLE-BOARD 6802 COMPUTER       *
                ************************************

                * DEFINE MEMORY ADDRESSES FOR COMPUTER
  (F800)        ROM    EQU   $F800
  (0000)        RAM    EQU   $0000
  (007F)        STACK  EQU   $007F
  (A000)        PIA1   EQU   $A000
  (A000)        PIADRA EQU   PIA1
  (A001)        PIACRA EQU   PIA1+1

                * RAM DATA
  (0000)               ORG   RAM
0000            CHAR   RMB   1          CURRENT CHARACTER BEING TRANSMITTED
0001            BITCTR RMB   1          COUNTER FOR 8 BITS PER ASCII CHARACTER

                * MAIN PROGRAM

                * START AND INITIALIZATION
  (F800)               ORG   ROM
F800 8E 007F    START  LDS   #STACK     SET STACK POINTER TO TOP OF RAM
F803 7F A001           CLR   PIACRA     RESET PIA CONTROL REGISTER
F806 86 FF             LDA A #$FF
F808 B7 A000           STA A PIADRA     SET DIR REG A FOR OUTPUT
F80B 86 04             LDA A #$04
F80D B7 A001           STA A PIACRA     TURN ON BIT 2 OF CONTROL REG

                * SEND FIVE SECONDS OF 1 (MARK) BEFORE STARTING
F810 CE 0226    WAIT   LDX   #550       FIVE SEC TIMES 110 BITS PER SEC
F813 8D 2B      WAIT1  BSR   MARK       SEND OUT A 1
F815 09                DEX
F816 26 FB             BNE   WAIT1      AND REPEAT 550 TIMES
F818 CE F870           LDX   #TEXT      POINT INDEX REGISTER TO TEXT

                * NOW SEND OUT THE NEXT CHARACTER
F81B A6 00      GO     LDA A 0,X        GET THE NEXT CHARACTER
F81D 97 00             STA A CHAR       AND SAVE IT
F81F 81 04             CMP A #$04       IS THIS THE END OF TEXT?
F821 27 ED             BEQ   WAIT       YES -- WAIT AND REPEAT
F823 86 08             LDA A #8         GET READY TO COUNT 8 BITS
F825 97 01             STA A BITCTR
F827 8D 2F             BSR   SPACE      SEND OUT START PULSE (0 = SPACE)
                * MAIN SHIFT LOOP TO SEND OUT NEXT BIT
F829 74 0000    SHIFT  LSR   CHAR       MOVE NEXT BIT INTO CARRY
F82C 24 04             BCC   SEND0      SEND OUT 0 IF CARRY=0
F82E 8D 10             BSR   MARK       OUTPUT A 1
F830 20 02             BRA   COUNT8
F832 8D 24      SEND0  BSR   SPACE      OUTPUT A 0
F834 7A 0001    COUNT8 DEC   BITCTR     HAVE WE SENT 8 BITS?
F837 26 F0             BNE   SHIFT      NO -- GO BACK TO SEND NEXT BIT
F839 8D 05             BSR   MARK       YES -- SEND OUT A STOP BIT
F83B 8D 03             BSR   MARK       AND ANOTHER STOP BIT
F83D 08                INX              POINT INDEX TO NEXT CHARACTER
F83E 20 DB             BRA   GO         AND GO BACK TO SEND NEXT CHAR

                * SUBROUTINE TO SEND OUT A MARK (1)
                * SEND OUT 20 CYCLES OF 2225 HZ; EACH HALF-CYCLE
                * IS 201 MACHINE CYCLES AT A .8947 MHZ CLOCK FREQ.

F840 C6 14      MARK   LDA B #20        SEND 20 CYCLES
F842 86 01      MARK1  LDA A #1
F844 B7 A000           STA A PIADRA     POSITIVE HALF-CYCLE
F847 86 20             LDA A #32        SET UP WAIT LOOP
F849 4A         MARK2  DEC A
F84A 26 FD             BNE   MARK2      WAIT
F84C 7F A000           CLR   PIADRA     NEGATIVE HALF-CYCLE
F84F 86 1F             LDA A #31        SET UP WAIT LOOP
F851 4A         MARK3  DEC A
F852 26 FD             BNE   MARK3      WAIT
F854 5A                DEC B            NEED MORE CYCLES?
F855 26 EB             BNE   MARK1      YES -- GO BACK
```

Let's assume that we want to set up port A so that five lines are inputs and three are outputs. The initialization sequence is shown in Listing 1, which first clears the control register, thereby making sure that bit 2 is 0 so the next step will load 00000111 into the direction register to set five bits for input and three for output. Once this is done, storing hex 04 into the control register sets bit 2 so that succeeding instructions will affect the data register. (The first step is not really necessary, but is a good safety precaution to make sure the control register is reset, just in case the RESET signal did not do a good enough job.)

A Complete Program

Now that we have looked at the various aspects of programming our Kilobaud Klassroom Komputer, it's time to put them all together and look at some complete programs. Let's start with the ACTESTER program which appeared in Kilobaud Klassroom No. 20 (September 1980 issue of *Kilobaud Microcomputing*, p. 85.) This program is reprinted here as Listing 2.

The object of this program is to output audio to an acoustic coupler (via a speaker or earphone), which can be decoded by the coupler and displayed on a terminal or printer. Though this program occupies less than a tenth of the 2716 EPROM on our computer, it has a lot of different functions and is an excellent demonstrator of computer programming.

The program outputs a message (which reads "THIS IS THE KILOBAUD KLASSROOM 6802 SINGLE-BOARD COMPUTER . . .") at 110 baud, the speed normally used by mechanical teleprinters. Fig. 3 shows what a typical ASCII character of this message looks like; this is the letter A, which has the hex code 41, or binary 01000001.

As described in earlier installments, serial data sent to a terminal usually consists of a start pulse (which is always 0), the eight data bits (sent backward) and one or two stop bits (which are always 1). Between characters, 1 is sent. In Fig. 3 we have

✓ *Reader Service—see page 226*

```
F857 39              RTS             NO -- SO EXIT

               * SUBROUTINE TO SEND OUT A SPACE (0)
               * SEND OUT 18 CYCLES OF 2025 HZ; EACH HALF-CYCLE
               * IS 221 MACHINE CYCLES AT A .8947 MHZ CLOCK FREQ.

F858 C6 12     SPACE  LDA B #18       SEND 18 CYCLES
F85A 86 01     SPACE1 LDA A #1
F85C B7 A000          STA A PIADRA    POSITIVE HALF-CYCLE
F85F 86 24            LDA A #36       SET UP WAIT LOOP
F861 4A        SPACE2 DEC A
F862 26 FD            BNE   SPACE2    WAIT
F864 7F A000          CLR   PIADRA    NEGATIVE HALF-CYCLE
F867 86 22            LDA A #34       SET UP WAIT LOOP
F869 4A        SPACE3 DEC A
F86A 26 FD            BNE   SPACE3    WAIT
F86C 5A               DEC B           NEED MORE CYCLES?
F86D 26 EB            BNE   SPACE1    YES -- GO BACK
F86F 39               RTS             NO -- SO EXIT

               * TEXT TO BE PRINTED OUT
F870 0D        TEXT   FCB   $D,$A,7,0,0,0
F876 54               FCC   'THIS IS THE KILOBAUD KLASSROOM 6802 '
F89A 53               FCC   'SINGLE-BOARD COMPUTER'
F8AF 0D               FCB   $D,$A,7,0,0,0
F8B5 4E               FCC   'NOW IS THE TIME FOR ALL GOOD MEN TO '
F8D9 43               FCC   'COME TO THE AID OF THEIR COUNTRY.'
F8FA 0D               FCB   $D,$A,0,0,0
F8FF 54               FCC   'THE FOX JUMPED OVER THE LAZY DOG'
F91F 0D               FCB   $D,$A,0,0,0
F924 30               FCC   '0123456789!"#$%&'
F934 0D               FCB   $D,$A,0,0,0
F939 04               FCB   4

               * 6802 RESET VECTOR AT FFFE-FFFF POINTS TO START

   (FFFE)             ORG   $FFFE
FFFE F8 00            FDB   START

                      END
```

a 1 before the character, then start with a 0 start pulse, follow with the bits 10000010 (which is the code for an A but sent with the least significant bit first) and end with two stop bits of 1. Two stop bits are always used at 110 baud, whereas higher speeds generally use only a single stop bit.

Since exactly 11 bits—counting the start bit, eight data bits and two stop bits—are sent for each character, sending data at 110 baud (which is 110 bits per second in this particular case) means that each character takes exactly 1/10 second, or 100 ms. Dividing this by 11 gives us a time of 9.09 ms for each bit of the character.

In this application, however, we want to send ASCII characters, not as digital pulses, but as tones which can be received by an acoustic coupler. An acoustic coupler is a device that might be used between a terminal and a telephone line to allow the terminal to talk with a computer at the other end of the telephone connection.

Since digital pulses cannot be sent on a telephone line, the bits are encoded as tones. Different tones are used in each direction so that data can be sent in both directions at the same time. The job of the acoustic coupler is to generate and receive these tones and couple them to the telephone acoustically.

Most often, the terminal end of the connection places the call to the computer; hence the acoustic coupler at the terminal is called an originate device, while the computer has an answer coupler. (The computer is generally directly, rather than acoustically, coupled to the phone line, and the coupler is called a modem rather than an acoustic coupler.)

The originate coupler encodes its data by sending a tone of 1270 Hz for 1 and 1070 Hz for 0; the answer modem sends its data using a tone of 2225 Hz for 1 and 2025 Hz for 0. So we must generate 9.09 ms of 2225 Hz tone for 1, and 9.09 ms of 2025 Hz tone for 0.

At 2225 Hz, each cycle of tone takes 1/2225 second, which means there is time for about 20 cycles in the 9.09 ms devoted to one data bit.

Likewise, at 2025 Hz a cycle of tone takes 1/2025 second, which allows time for about 18 cycles in the 9.09 ms devoted to one data bit.

The program sends the tone out one of the PIA bits as a square wave, simply by turning the bit on and off. Though the resulting tone is a square wave rather than a sine wave, it is generally rounded off by an inexpensive speaker or earphone into something resembling a sine wave, so that the acoustic coupler will accept it quite well anyway.

The timing of the tone is handled by the 6802 processor. Since we are running the 6802 from a 3.579 MHz crystal, which results in a clock frequency of one-quarter of that, the resulting clock frequency of 0.8947 MHz makes each machine cycle about 1.118 ms long. At the 2225 Hz tone, a half-cycle lasts 1/4450 second, or 225 ms. Dividing this by the cycle time of 1.117 ms tells us that a new half-cycle of tone must be sent out every 201 or so machine cycles. Repeating the same calculation for the 2025 Hz tone gives a time of about 221 machine cycles.

Thus, the generation of the audio tones to be output is completely handled by the microprocessor—counting the number of machine cycles (either 201 or 221) sets the frequency of the tone, while generating a specific number of audio cycles (either 20 or 18) sets the duration of that tone.

All of this is handled by two subroutines within the program called Mark and Space. By tradition, the 1 signal is called a mark, so the Mark subroutine sends out 9.09 ms of 2225 Hz tone; the 0 signal is called a space, so the Space subroutine sends out 9.09 ms of 2025 Hz tone.

With this as a background let's examine Listing 2 to see just how the program works.

The first part of the program tells the assembler what addresses to use for various components. In the basic computer, the 2716 EPROM occupies addresses F800 through FFFF while RAM occupies addresses 0000 through 007F. Therefore we define ROM at $F800 and RAM at $0000.

Since the stack is filled from

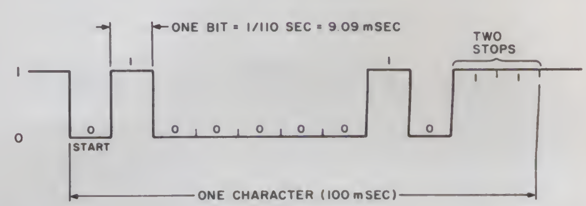


Fig. 3. An ASCII character (the letter A, hex 41) at 110 baud.

```
WAIT    LDX  #550    Load 550 into index
WAIT1   BSR  MARK    Send out 2225 Hz
        DEX          Decrement index
        BNE  WAIT1   Repeat if not 0
```

Listing 3.

the top down, we define the stack at location $007F, which is the top of the RAM in the system. PIA1 is addressed at location $A000, with the symbol PIADRA standing for PIA data/direction register A and PIACRA standing for PIA control register A. These two are located at addresses A000 and A001.

The next part of the program sets up space for two variables at the beginning of RAM. The statement ORG RAM tells the assembler to start allocating memory from location 0000 (since RAM had previously been defined to EQUal 0000). CHAR RMB 1 sets aside one location for the variable CHAR, which will hold the character currently *being transmitted.* BITCTR RMB 1 sets aside another location for a counter, which will be used to count eight bits.

Now follows the main program. The statement ORG ROM tells the assembler to assign space for it beginning at location ROM, which had previously been defined as address F800.

The instruction START LDS #STACK loads the stack pointer *with 007F,* thus telling it to allocate the RAM area from 007F down for the stack. (When you write your own programs for the 6802 computer, don't forget to initialize the stack pointer register to a free area of RAM as one of your first instructions; otherwise, you will be at a loss to figure out why the program doesn't work!)

The next four instructions initialize the PIA as follows:

```
CLR    PIACRA
LDA A  #$FF
STA A  PIADRA
LDA A  #$04
STA A  PIACRA
```

The first instruction makes sure that bit 2 of the control register is 0 so the program can load the direction register. The next two steps load a binary 11111111 into the direction register, thus making every bit of port A an output; the program only uses one bit for output, and initializing the others is done for no particular reason. Finally, storing 04 into the control register sets bit 2 and makes address A000 apply to just the data register.

Before sending out data, the output should be a constant 1. The next part of the program sends out five seconds of 2225 Hz tone. Since there are 110 bits per second, this means that 550 bits must be sent out. This is done in a loop as shown in Listing 3. Here the index register is used as a counter to execute the Mark subroutine 550 times while the index register counts down from 550 to 0.

When five seconds of 2225 Hz has been sent out, the instruction LDX #TEXT loads the index register with the address of the Text string which is at the end of the program.

The Text string is defined at the end of the program by a series of statements beginning with

```
TEXT   FCB $D,$A,7,0,0,0
```

The character $0D is a carriage return, $0A is a line feed, and $07 rings the bell; this is equivalent to BASIC's

```
TEXT$ = CHR$(13) + CHR$(10) + CHR$(7) +
CHR$(0) + CHR$(0) + CHR$(0) + . . .
```

and will eventually cause the terminal to go to the next line, ring the bell and print three nulls (code 0) which are needed by teleprinter machines to give them a bit of extra time to finish doing the carriage return prior to printing on the next line.

The next line of the text uses the FCC directive to tell the assembler to insert the ASCII codes for the letters THIS IS THE. . . . The text is broken up among several FCC statements, with FCBs inserted between lines to supply carriage returns and line feeds. Finally, the FCB 4 inserts a 04 code at the very end; the program will test for this to determine when the string ends. Thus, these 12 lines near the end of Listing 2 define a long string of ASCII text in memory ending with 04; the instruction LDX #TEXT at location F818 simply points the index register to the beginning of this string in preparation for sending out the next character.

In the next part of the program, GO LDA A 0,X is an indexed load instruction. The 0 near the end is an offset which is added to the address in the index register. Adding zero obviously does nothing, so the address in the index register is used exactly as is. Since it is pointing at the first character in the Text string, this instruction tells the computer to load that character into accumulator A. The STA A CHAR instruction following it tells the computer to store that character at location CHAR.

The next two instructions

```
CMP A  #$04
BEQ    WAIT
```

tell the computer to compare the character just picked up with a 04 code and branch to WAIT if it is equal to 04. This is the test which stops output when the 04 character at the end of the Text string is encountered and makes it restart.

Next, location BITCTR is initialized to 8 with

```
LDA A  #8
STA A  BITCTR
```

and will be used to count out eight bits. This is followed by a BSR SPACE instruction, which goes to the Space subroutine to send out 9.09 ms of 0 tone at 2025 Hz to act as a start pulse.

We finally get to the main loop, which sends out the eight bits of the character. It begins with

```
SHIFT LSR CHAR
```

The LSR instruction is a logical shift right. (See Table 1, the instruction set table, in Kilobaud Klassroom No. 22, on page 71 of the November 1980 issue of *Kilobaud Microcomputing.* Look at the right side of the table, where a picture similar to Fig. 4

shows the operation of that instruction.) In this instruction, each bit of the location CHAR is shifted right one place; the rightmost bit, bit 0, goes into the carry bit (in the condition code register), while the leftmost bit, bit 7, is filled with 0.

The very first character is a carriage return, or the hex number 0D. This is shifted as follows:

	Location CHAR		Carry
Before:	0 0 0 0 1 1 0 1		-
After:	0 0 0 0 0 1 1 0	-->	1

Since we want to send out the eight bits of the character backward—that is, the least significant bit first—this moves that very same bit into the carry bit and removes it from CHAR.

The next four lines test the carry and send out a space if it is 0, or a mark if it is 1:

```
        BCC  SEND0
        BSR  MARK
        BRA  COUNT8
SEND0   BSR  SPACE
COUNT8...
```

If the carry bit is 0, the BCC (branch if carry clear) instruction sends the program to SEND0 to send out a space; otherwise, it sends out a mark (both of these are sent out by subroutines that it goes to with the BSR [branch to subroutine] instruction).

Fig. 4. The LSR instruction.

The next two instructions are

```
COUNT8 DEC  BITCTR
       BNE  SHIFT
```

The first decrements the bit counter, and the second sends the program back to SHIFT, as long as the result is not zero. The result is a loop which shifts CHAR eight times to the right, and sends out the character one bit at a time, starting from the right.

Finally, the computer performs two BSR MARK instructions to send out two stop bits. This completes the entire character, so an INX increments the index register to point to the next character in the string, and a BRA instruction branches back to GO to repeat the same process for the next character.

That completes the main body of the program, leaving us to explain only the Mark and Space subroutines. Except for different numbers, the two subroutines are identical. In fact, the same subroutine could have been used with just different initialization, but I chose to use separate ones to make the program easier to understand.

To send out a mark of 2225 Hz, we send out 20 cycles of tone, where each half-cycle lasts for about 201 machine cycles. The subroutine does this with three loops. An outer loop is repeated 20 times, with accumulator B acting as the counter. The two inner loops, one for each half-cycle, are repeated for 32 and 31 repetitions, respectively, and use accumulator A as the loop counter. The loop starts with

```
MARK    LDA  B  #20
MARK1   LDA  A  #1
        STA  A  PIADRA
```

where the B accumulator is initialized at 20, while the A accumulator is used to send 1 out to the rightmost bit of the PIA port A. This starts the positive half-cycle of the tone (the tone output is taken from bit 0 of port A, and is +5 volts on the positive half-cycle, and 0 volts on the negative half-cycle).

The next three instructions set up a wait loop as follows:

```
        LDA  A  #32
MARK2   DEC  A
        BNE     MARK2
```

This time accumulator A is the counter. It is decremented, and the BNE instruction loops back to decrement A again until it reaches zero.

As the instruction set tables in the November installment show, the DEC A instruction takes two machine cycles, while the BNE takes four. The loop, therefore, takes six cycles, and thus 32 repetitions require 192 machine cycles total.

As you remember, we calculated that we need 201 cycles; the loop runs a bit less than that to give some extra time for those instructions outside the loop. (Actually, the program runs just a bit slow; it could have been fine-tuned to provide exactly the right time delays, but such care is not needed in this application.)

The next four instructions repeat the process, but this time sending out 0 to the PIA and looping 31 times for a total of 186 machine cycles. As before, some extra time is allotted to instructions outside the loop.

The subroutine ends with

```
DEC  B
BNE     MARK1
RTS
```

where the B accumulator is decremented, and as long as it is not zero the program goes back to send out another positive half-cycle; otherwise, the RTS is like BASIC's RETURN to exit from the subroutine.

6802 Assembler Notes

There are several lines in the above program which will not work on the 6802 assembler we used last time.

First, this assembler will not like the plus sign in

```
PIACRA EQU   PIA1+1
```

Instead, you will have to define it more exactly as

```
PIACRA EQU   $A001
```

Second, this assembler does not like multiple bytes within a single FCB statement. Hence, the line

```
TEXT   FCB $D,$A,7,0,0,0
```

will have to be broken down into

```
TEXT   FCB  $D
       FCB  $A
       FCB  7
       etc.
```

Other than that you should have no trouble assembling this program.

Next Month

Kilobaud Klassroom No. 25 will be the last article in this series. We will develop an entire burglar alarm controller around our single-board computer. By way of comparison, look at the "Computerized Security and Status System" by Richard R Parry on page 30 of the November 1980 issue of *Kilobaud Microcomputing*. That system could run exactly as described on the Kilobaud Klassroom Komputer. ■

WE WILL NOT BE UNDERSOLD

DISK DRIVES $314

40 track, 102K Bytes. Includes power supply and TRS-80* compatible silver enclosure. Ready to plug-in and run the moment you receive it. Can be intermixed with each other and Radio Shack drive on same cable. 90 day warranty. One year on power supply. Available for 220 Vac (50 Hz) operation. **External card edge included.**

FOR TRS-80*		
CCI-100	5¼", 40 Track (102K Bytes) for Model I	$314
CCI-280	5¼", 80 Track (204K Bytes) for Model I	$429
CCI-800	8" Drive for Model II (½ Meg Bytes)	$795
For Zenith Z89		
CCI-189	5¼", 40 Track (102K Bytes) add-on drive	$394
Z-87	Dual 5¼" add-on drive system	$995

DISKETTES — Box of 10 (5¼") - with plastic library case

Maxell $30 **BASF or Verbatim** $24
8" double density for Model II (box of 10) $36
CLEAR PLASTIC CASE - Holds 50 diskettes $19

DISK OPERATING SYSTEMS

PATCHPAK #4 by Percom Data	$ 8.95
CP/M** for Model I, Zenith **$145** • for Model II, Altos	$169.00
NEWDOS Plus **40track**	$ 79.00
NEWDOS 80	$135.00

COMPLETE SYSTEMS

ALTOS 64K, DD, SS, 2-Drive, 1MB **ACS 8000-2**	$3395
APPLE 16K	$969
TRS-80* Model II-64K	$3499
TRS-80* Model III - 16K	$899
TRS-80* Expansion Interface	$249
ZENITH Z89, 48K all-in-one computer	$2440
ZENITH Z19	$735
TELEVIDEO 920C	$748
ATARI 400 $489 **ATARI 800**	$747
APF Game Only $ 95 **Complete System**	$489
MATTEL INTELLIVISION	$229

MONITORS

LEEDEX	12" B & W Video 100	$129
ZENITH	13" Color	$379
SANYO	9" B & W VM4509	$155
SANYO	12" B & W DM5012	$210
SANYO	12" Green Screen DM5112	$215
SANYO	13" Color DMC6013	$375
APF	9" B & W TVM-10	$139

TELECOMMUNICATIONS

LIVERMORE STAR MODEM 2-year guarantee	$145
CAT MODEM Works same as Radio Shack Telephone Interface II	$148
D-CAT HARD WIRED DIRECT MODEM	$189

COMMUNICATIONS SOFTWARE

CCI-TELNET VERSION 5: A communication package which enables microcomputer users to communicate both with large mainframes and other microcomputers. Completely CP/M compatible. Multiple communication protocols supported. **$149**

INTELLIGENT TERMINAL SYSTEM ST-80 III: Enables a TRS-80* to act as a dial-up terminal on any time sharing network. **$139**

16K MEMORY UPGRADE KITS 2 for $65 $35

200 ns for TRS-80*, Apple II, (specify): **Jumpers** **$2.50**

PRINTERS

NEC Spinwriter

Letter Quality High Speed Printer
Includes TRS-80* interface software, quick change print fonts, 55 cps, bidirectional, high resolution plotting, graphing, proportional spacing: R.O. **$2395**
R.O. with Tractor Feed **$2575** KSR with Tractor Feed **$2950**

C.ITOH Starwriter, 25 CPS, daisy wheel printer **$1895**
C.ITOH Starwriter II, 45 CPS, daisy wheel printer **$2195**
Letter quality printers. Use up to 15" paper. 1 year warranty on parts. 3 months on labor. Proportional spacing and bidirectional printing. Same as VISTA V300.

779 CENTRONICS TRACTOR FEED PRINTER Same as Radio Shack line printer I	$969
737 CENTRONICS FRICTION & PIN FEED PRINTER n × 9 proportional and 7 × 8 mono spacing. Same as Radio Shack line printer IV	$795
730 CENTRONICS FRICTION & PIN FEED PRINTER 7 × 7 matrix Same as Radio Shack line printer II	$595
P1 CENTRONICS PRINTER Same as Radio Shack quick printer	$269
PAPER TIGER (IP445)	$699
(IP440) Includes 2K buffer and graphics option	$879
(IP460) Bidirectional, 160 cps, graphics and 2K buffer	$1075
TI-810 Faster than Radio Shack line printer III. Parallel and serial w/TRS-80* interface software w/u + l case & paper tray	$1589
Compressed print, vertical form control	$1865
OKIDATA Microline 80 Friction and pin feed	$545
Tractor Feed, friction, and pin feed	$645
Microline 82 Bidirectional, friction and pin feed	$745
Microline 83 Bidirectional, 120 cps, uses up to 15" paper	$1050
EATON LRC 7000 + 64 columns, plain paper	$289
ANADEX **DP-9500/01 $1350** **DP-8000**	$795

ACCESSORIES

HEAD CLEANING DISKETTE: Cleans drive Read/Write head in 30 seconds. Specify 5¼" or 8". **$20 ea/$45 for 3**

FLOPPY SAVER: Protection for center holes of 5¼" floppy disks. Installation tools and rings for 25 diskettes. **$ 11.95**
Re-orders of rings only **$ 6.95**

EXTERNAL DATA SEPARATOR: Eliminates data separation problems (crc). Improves reliability. This plug in unit comes fully assembled and tested. **$ 29.95**

Z-80 SOFTCARD: Your key to software expansion. The plug-in Z-80 Softcard transforms your Apple into a Z-80 while keeping all the benefits of the 6502. Comes with CP/M in two disk format, MBASIC and GBASIC, full documentation and utility programs. **$339**

RF MODULATOR: Adapts video to TV	$ 35.00
TRS-80 & OTHER MYSTERIES	$ 18.95
NEC SPINWRITER THIMBLE **$11.95** **RIBBON**	$ 6.00
CCS CARDS: Parallel or serial printer interface cards	$115.00
RS232: For Radio Shack Interface.	$ 84.00
TRS232: Teletype current loop output from cassette port	$ 49.00
DISK-DRIVE EXTENDER CABLES: Fits all mini-disk drives.	$ 16.95
SIX (6) PRONG ISOLATOR: ISO-2	$ *54.00*
AC FILTER/6 PRONG POWER STRIP	$ 39.00
DISK DRIVE CABLES: 2 drive **$29.00** 4 drive	$ 35.00
DUST COVERS: TRS-80/Apple	$ 7.95
PLASTIC DISKETTE HOLDER: For ring binder, holds 20	$ 8.00

For fast delivery, send certified checks, money orders or call to arrange direct bank wire transfers. Personal or company checks require two to three weeks to clear.

DEALER (NATIONAL/INTERNATIONAL) INQUIRIES INVITED **Send for FREE Catalogue**

The CPU SHOP ✓256

TO ORDER CALL TOLL FREE 1-800-343-6522
TWX: 710-348-1796 Massachusetts Residents call 617/242-3361

5 Dexter Row, Dept. K01M
Charlestown, Massachusetts 02129
Hours 10AM-6PM (EST) Mon.-Fri. (Sat. till 5)

Massachusetts Residents add 5% Sales Tax
*TRS-80 is a Tandy Corporation Trademark
** Digital Research

master charge THE INTERBANK CARD
VISA*

✓ Reader Service—see page 226

For about $10 you have your choice of reverse video—full-screen or fully programmable.

Reverse Video For the OSI C1P

By Donn Burke Baker

Reverse video—black characters on a white background—has several advantages.

For example, characters are often more distinct than with normal video, especially on inexpensive monitors or with systems that use a TV set and rf modulator.

Also, if the reverse function is programmable, areas of the display can be made to stand out dramatically for operator attention, to emphasize a critical figure, or simply to indicate that input is required.

Unfortunately, even full-screen reverse video is seldom available on small systems, and a programmable reverse video is usually found only on sophisticated systems or intelligent terminals.

An article in the December 1979 issue of *Microcomputing* (''Reverse Video from OSI's 540 Board'') describes a simple method of obtaining full-screen reverse video for OSI's 540 video board. Upon examination of my C1P's manual, I discovered that, electrically, the circuits are identical (Ohio Scientific, 1978). The video output portion of the C1P display circuit is shown in Fig. 1. After reading the article, I jumpered the final video inverter (U70, pins 4, 5 and 6) with a miniature toggle switch brought out to the front panel.

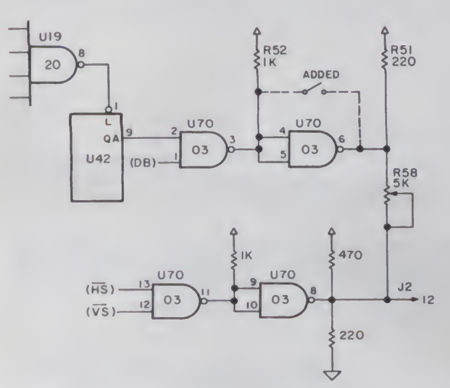


Fig. 1. Video output circuit—OSI C1P.

But my full-screen reverse video had one serious problem. I needed to adjust both the contrast and brightness controls of my monitor considerably for an acceptable quality. To make the matter worse, I couldn't adjust the monitor so that the quality of both the normal and reverse video displays were acceptable upon switching.

This set of hardware modifications for the Ohio Scientific C1P provides either a full-screen reverse video or fully programmable reverse-video display of high quality. The hardware costs are about $10.

Full-Screen Reverse Video

I found two flaws in the original method of obtaining reverse video. First, connection of the inverter output to its input (through the toggle switch) leaves that circuit open to high-frequency oscillations. Because of the fast switching time of the 74LS03 used as an inverter (typically less than 15 ns), a wire length to the switch of about 20 inches could allow oscillations. In turn, enough power could be drawn to damage the 74LS03, as well as cause interference with some radios and TVs.

The second problem is the change in video drive levels caused by paralleling R52 and R51 with the toggle switch. (See Fig. 1.) While you can compensate for this by adjusting R58, such adjustment precludes switching casually between normal and reverse modes of video display. It also prevents the point-by-point selection of normal or reverse video.

To restore the proper drive level, remove the possibility of high-frequency oscillation and permit programmable reverse video, I needed an electronic switch for the selection of reverse/normal video. Such a switch is the exclusive-OR (*XOR*) illustrated in Fig. 2a. While usually used in the logical design of circuits such as arithmetic adders, the XOR has the property of passing a signal or its complement, depending on the second, or control, input.

As can be seen from the truth table (Fig. 2b), input B can be thought of as the control signal, and input A as the desired signal. When B is logic 1 (reverse), then the complement of A is transferred to the output. When input B is logic 0 (normal), the A signal

Donn Burke Baker, 3128 Silver Lake Rd., Minneapolis, MN 55418.

is transferred unchanged.

Note that the circuit can be constructed from four two-input NAND gates. While XOR circuits can be purchased, I used discrete NAND gates. Only three gates are required since the fourth gate, the output, is the 74LS03 that is part of the C1P video output circuit. The other three gates are obtained by adding a 74LS00 to the prototype IC position at U44 on the C1P board. Fig. 2c shows the final implementation of the XOR in the C1P.

Photo 1 shows a portion of the C1P circuit board. IC position U44 is along the bottom edge, near center, just to the right of the crystal. U70 is located at the extreme left, along the bottom. Pin 1 of both ICs is toward the upper right of the photograph.

Install a 74LS00 at U44. A socket is not required, but recommended. The location has pin pads for a 16-pin IC, while the 74LS00 has only 14 pins. The pads closest to the crystal should be left unused. Be sure to install the IC (and socket, if used) with pin 1 toward the center of the board, like the other ICs in the row.

Locate U70, pins 3, 4 and 5. Note that all three are connected by small printed circuit (PC) runs between the pin pads. Pin 3 also has a PC run to R52.

Carefully cut the PC runs between pins 3 and 4 and between 4 and 5. (Do not disturb the PC run from pin 3 to R52.) Completely remove the cut runs from between the pin pads. (I recommend a new knife blade like X-acto for these operations.)

Connect the +5 volt bus (which runs across the top of position U44) to U44, pin 14, and the GND bus (running along the bottom of U44) to pin 7. Connect U70, pin 3, to U44, pins 1 and 13. Connect U70, pin 4, to U44, pin 11; and U70, pin 5, to U44, pin 6. Connect U44, pins 3, 4 and 12, together. All of these connections should be short and as direct as possible.

Connect U44, pins 2 and 5, together. If the programmable reverse video feature yet to be described will not be added, wire a toggle switch between U44, pin 2, and ground. The unused pin pads at the bottom of the location make good tie points for leads to the switch. If the programmable reverse video will be added, the switch is only temporary.

All soldering should be done with a fine-tip soldering iron. All added wires should be of 28 or 30 AWG gauge insulated wire. Also be careful

While some minor adjustment of contrast and brightness controls might be necessary, you should not need to make adjustments when switching between normal and reverse modes.

not to strip too much insulation from the wire ends, since this can cause shorts between IC pins or PC runs. Be careful not to short pins and runs with solder bridges. Use solder very sparingly.

At this point, full-screen reverse video is available. Double-check all work and apply power if everything is OK. With the toggle switch closed (grounding U44, pins 2 and 5), a normal video display is produced. Opening the switch, letting U44, pins 2 and 5, go to +5, causes the video signal to be complemented by the XOR circuit, producing a quality reverse video display.

While some minor adjustment of contrast and brightness controls might be necessary, you should not need to make adjustments when switching between normal and reverse modes. No adjustment of R58 is required, if properly set in the first place.

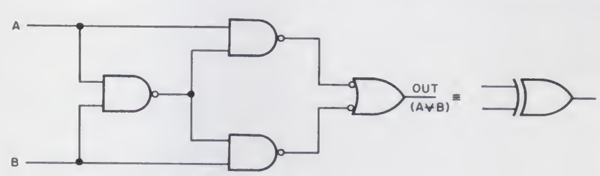
Programmable Reverse Video

Before going into the details of the programmable reverse video feature, I'll give a brief description of how the current video circuit works.

Like many microcomputer systems, the C1P uses a portion of its memory address space as a video display memory. Since the display on the C1P is 32 columns by 32 rows, 1024 addresses are required. The range of these addresses is D000–D3FF (hex) or 53248–54271 (decimal).

To display a character on the video display, place the eight-bit code for that character at the memory address location corresponding to the desired location on the display screen. This may be done by an STA instruction in 6502 assembly language or by a POKE in BASIC. Special address counters and multiplexers continuously present successive memory locations within the display memory address range to a ROM character generator. This generator translates the eight-bit code into the video signal necessary to control the monitor display. The video is amplified, combined with horizontal and vertical sync signals and sent to the monitor.

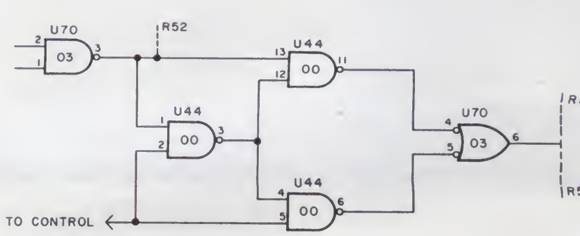
In designing the programmable reverse video, I decided that four requirements were to be met: any point, or combination of points, on



(A) EXCLUSIVE-OR

B	A	A⊻B
0	0	0
0	1	1
1	0	1
1	1	0

(B) TRUTH TABLE



(C) AS IMPLEMENTED

Fig. 2. Electronic reverse video switch.

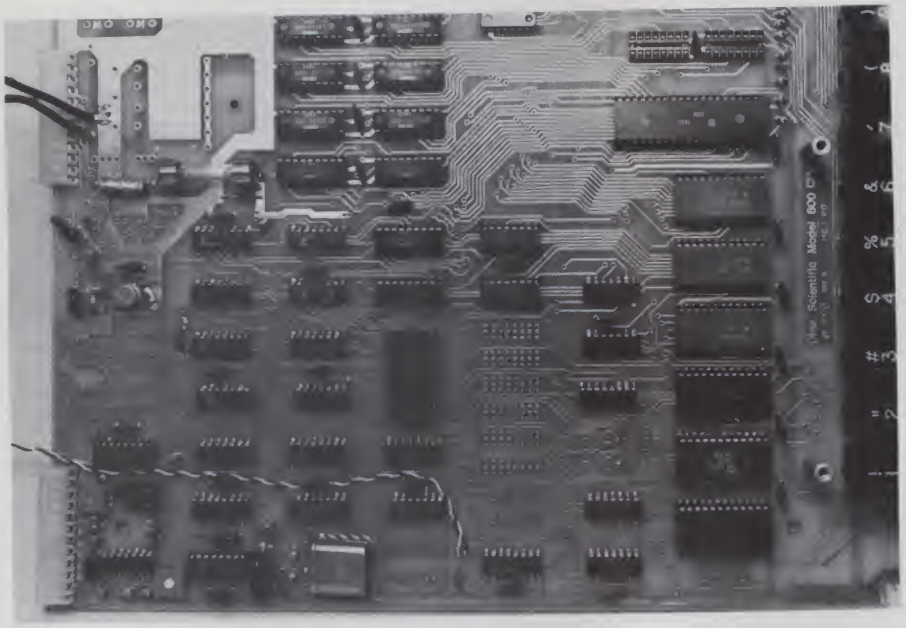


Photo 1. Portion of the C1P circuit board before modification.

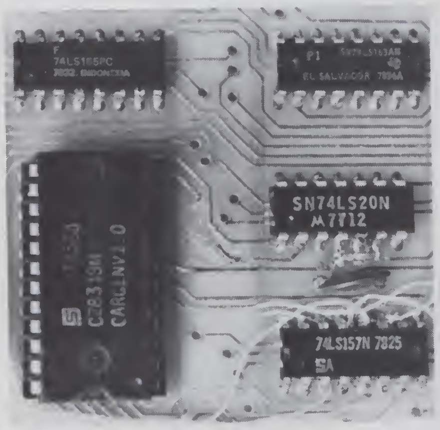


Photo 2. PC run from U56, pin 2, cut and removed.

the display must be programmable; the programming of a point should not be affected by a screen scroll; locating a point should be simple (i.e., the software necessary to program a point should be simple); and modifications required must be kept to a minimum.

By adding a one-bit-wide memory in an unused portion of the memory space, normal/reverse control information may be programmed in a manner identical to that used to place a character at any position within the display. The size of this memory is the same as the display memory, 1024 locations, and if located contiguously with the upper end of the present display memory, the added memory can be addressed by a simple offset of 1024 from the address of the character position.

The memory allocation map for the C1P calls out D000–D3FF (hex) as the video memory. The next location, however, is at DF00—the polled keyboard. Thus, D400- D7FF is currently unused, and is the size required for the added memory. This is true of all present configurations for the C1P: C1P; C1P-8K; C1P-MF, with P-DOS; and C1P-MF with OS56D. The use of D400–D7FF (hex) or 54272–55296 (decimal) allows any character to be displayed in reverse video by poking a control code for reverse display.

To implement this memory, you need some additional hardware. By modification and use of portions of existing hardware, the actual additional hardware is limited to a single memory IC (a 21L02) and a dual-D flip-flop (74LS74). Fig. 3 shows the added components, and also the use of various parts of the original circuit currently unused.

All necessary address decoding for the reverse/normal memory may be derived from currently existing signals, without the addition of extra logic elements. Recall that the added memory space starts where the present display memory ends. The range of addresses is D000–D3FF for the display, and D400–D7FF for the control memory. In terms of the address bus, these addresses are ($A15 \cdot A14 \cdot \overline{A13} \cdot A12 \cdot \overline{A11} \cdot \overline{A10} \cdot$ *(address within 1024 bytes as set by A9–A0)) and ($A15 \cdot A14 \cdot \overline{A13} \cdot A12 \cdot \overline{A11} \cdot A10 \cdot$ (address within 1024 bytes set by A9–A0)). Note that only bit A10 changes, aside from the bits that select one-of-1024, bits A9–A0.

*The centered dot is the symbol for the logic operation AND.—Editors.

The high-order three bits of address (A15, A14, A13) are decoded by U23, a three-to-eight line decoder. The output corresponding to $A15 \cdot A14 \cdot \overline{A13}$ (named $(\overline{V})$ in OSI's documentation) is used as one input to a four-input NAND at U56. The remaining inputs of U56 are A12, $\overline{A11}$ and $\overline{A10}$. Output of this gate (named $(\overline{VA})$) switches the address lines of the display memory between the refresh counter and the memory bus.

It is obvious that address line A10 has become a "don't care." That is, since it assumes both the true and not true states for the address multiplexer, it may be omitted. This is done by simply removing $\overline{A10}$ from the input of the NAND gate at U56 (see Fig. 3). Carefully cut and remove a short section of the PC run to U56, pin 2, near the top of the IC location. Photo 2 shows the PC run cut away.

A memory enable ($\overline{CE}$) is developed from Ø2 and GND, selected by an unused portion of the address multiplexer, U55. Using a short jumper, connect U55, pin 13, to GND. Wire U55, pins 11 and 14, together. Connect U55, pin 12, to U28, pin 13, with a short, direct wire.

Install a 21L02 at U28. A socket is recommended. Wire U28, pin 10, to +5 and U28, pin 9, to GND.

The R/$\overline{W}$ signal for the added mem-

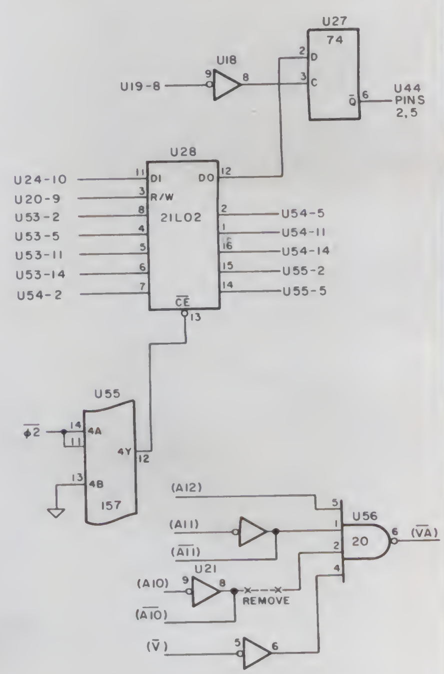


Fig. 3. Programmable reverse video switch.

$ DISCOUNT $

TRS-80® DEALER

MODEL II

26-4002
64K 1 Drive
$3440.00

26-4160 1 Drive EXP	$1035.00
26-4161 2 Drive EXP	1575.00
26-4162 3 Drive EXP	2115.00
26-4501 Gen. Ledger	180.00
26-4502 Inventory	180.00
26-4503 Payroll	360.00
26-4554 Acct. Rec.	180.00
26-4701 Fortran	270.00
26-1157A Daisy Wheel	2495.00
26-1158 Daisy Wheel II	1799.00

COMPUTER SPECIALISTS

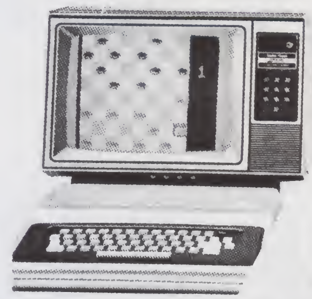
26-1155 Quick Printer II	$187.00
26-1145 RS-232 Board	84.00
26-1140 "O" K Interface	249.00
26-1141 "16" K Interface	359.00
26-1142 "32" K Interface	469.00
26-1160 Mini Disk - Drive O	419.00
26-1161 Mini Disk - Additional	419.00
26-1154 Lineprinter II	699.00
26-1156 Lineprinter III	1799.00
26-1159 Lineprinter IV	859.00
26-1166 Line Printer VI	1080.00
26-1563 Scripsit - Disk	79.00
26-1566 Visicalc	83.00
26-1562 Profile	72.00

NOTE: Call for availability of VIDEO TEX, Model III, Color, and other new products.

MODEL III

26-1061 4K I	$630.00
26-1062 16K III	888.00
26-1063 32K III 2-Drives, RS232	2225.00

COLOR



26-3001 4K	$360.00
26-3002 16K	540.00
26-3010 Color Video	360.00
26-1206 Recorder	54.00
26-3008 Joysticks	22.50

CENTRONICS

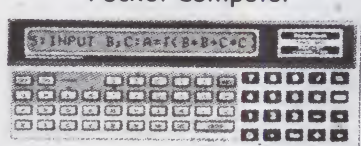
Fast 100 CPS Centronics
730 Printer.................. $659.00
Text Quality Centronics
737 Printer.................. $819.00

ALL OTHER R.S. SOFTWARE FURNITURE, STANDS, CABLES AND ACCESSORIES AT DISCOUNT FROM CATALOG PRICE.

Novation Cat Modem . . $149.00
CCA Data Management System 72.00
Adventure Games
Games 1-9 each 14.00

Model II Cobol Compiler
$360.00
Cobol Run Time Package
$36.00

Pocket Computer



26-3501 1.9K P.C.	$225.00
26-3503 Cassette I/F	45.00
14-812 Recorder	72.00

Acorn Software Products, Inc.

GAMES:

Alien Invasion	$9.00
Stock Market	9.00
Star Trek	9.00
Block 'Em	9.00
Ting-Tong	9.00

UTILITIES:

System Savers	14.00

EDUCATION:

Language Teacher	18.00

FREE: COMPUTER CATALOG UPON REQUEST

1-800-841-0860 Toll Free Order Entry

MICRO MANAGEMENT SYSTEMS, INC. ✓100

No Taxes on Out Of State Shipments

Immediate Shipment From Stock on Most Items

DOWNTOWN PLAZA SHOPPING CENTER
115 C SECOND AVE. S.W.
CAIRO, GEORGIA 31728
(912) 377-7120 Ga. Phone No.

R.S. 90 Day Limited Warranty
F-48 Form Provided

Largest Inventory In the S.E. U.S.A.

*TRS-80 is a registered trademark of the Tandy Corp.

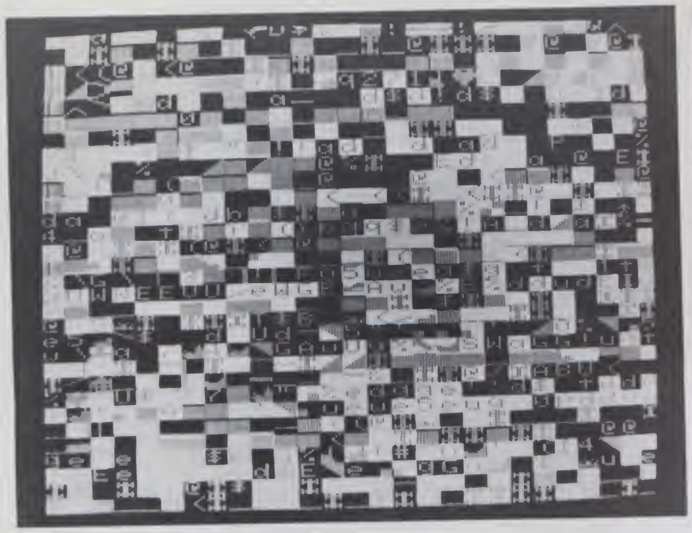


Photo 4a. A "power on" screen pattern. Note randomness of reverse video squares with this 21L02 memory.

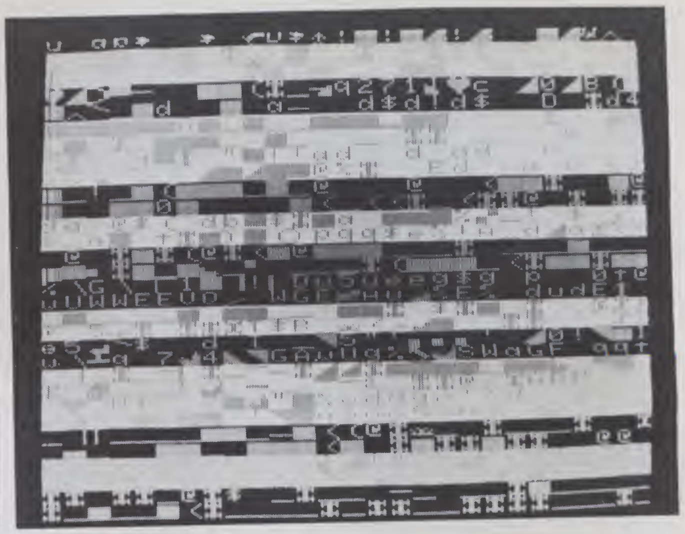


Photo 4b. Another "power on" screen pattern. With this 21L02, bands of reverse video are placed on the screen.

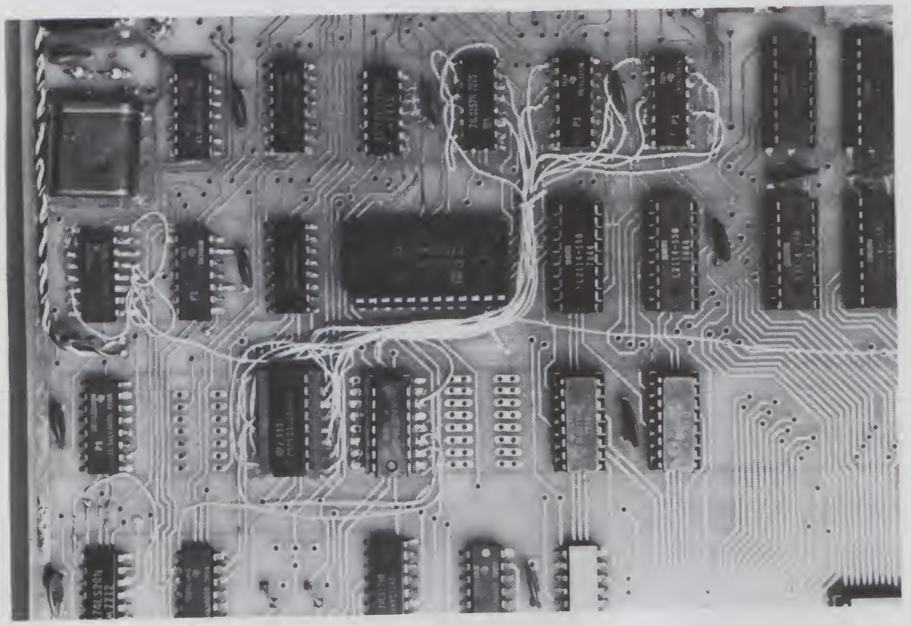


Photo 3. Completed programmable reverse video modifications. Note added ICs at U44, U27 and U28.

Photo 4c. Display response to "BREAK" keyin.

ory is taken from the decoder at U20 corresponding to an address greater than D3FF but less than D7FF ($A15 \cdot A14 \cdot \overline{A13} \cdot A12 \cdot \overline{A11} \cdot A10$). Using a short, direct wire, connect U20, pin 9, to U28, pin 3.

The data input signal for the added memory is taken directly from the data bus, bit 0. Connect U24, pin 10, to U28, pin 11.

The low-order ten bits of address, necessary to select one byte of 1024, are taken from the address multiplexers (U53, U54 and U55). These are controlled by the output of U56 ($\overline{VA}$). The address bits are wired from the input side of the multiplexer. Connect U53, pin 2, to U28, pin 8; U53, pin 5, to U28, pin 4; U53, pin 11, to U28, pin 5; and U53, pin 14, to U28, pin 6. Connect U54, pin 2, to U28, pin 7; U54, pin 5, to U28, pin 2; U54, pin 11, to U28, pin 1; and U54, pin 14, to U28, pin 16. Connect U55, pin 2, to U28, pin 15; and U55, pin 5, to U28, pin 9. Although lead dress is not critical, it will help to keep the added wires short and neatly dressed.

The last part of the modification is the type-D flip-flop shown in Fig. 3. This flip-flop serves to hold the reverse/normal control signal while the display is being generated. It also synchronizes the presentation of the reverse/normal control and character data being displayed.

Install the 74LS74 at U27. Again, a socket is recommended. Be sure to install pin 1 toward the center of the board and leave the bottom pin pads unused. Wire +5 and GND to pins 14 and 7, respectively. Connect U28, pin 12, to U27, pin 2.

The $\overline{Q}$ output of the flip-flop, U27, pin 6, is wired to the XOR circuit in place of the toggle switch used earlier. Disconnect the toggle switch from U44, pin 2, and wire U27, pin 6, to U44, pin 2, instead.

(The toggle switch *may be left in* the circuit. This would provide a full-screen reverse-video display (switch closed) or a programmable display (switch open).)

Only one control signal is needed for the flip-flop: a clock. This signal exists already to load the character generator signal into the serial shift register at U42. However, this signal is of the wrong polarity.

An inversion is required, or the reverse/normal control will be applied to the XOR one display column too

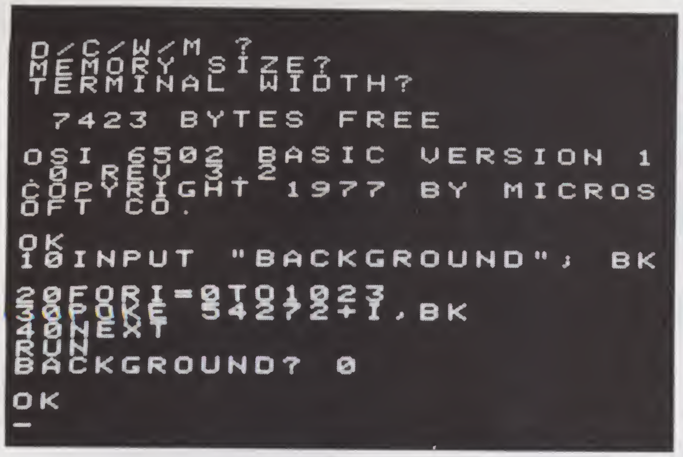


Photo 5b. Result of running the Checkout program for a "normal" display.

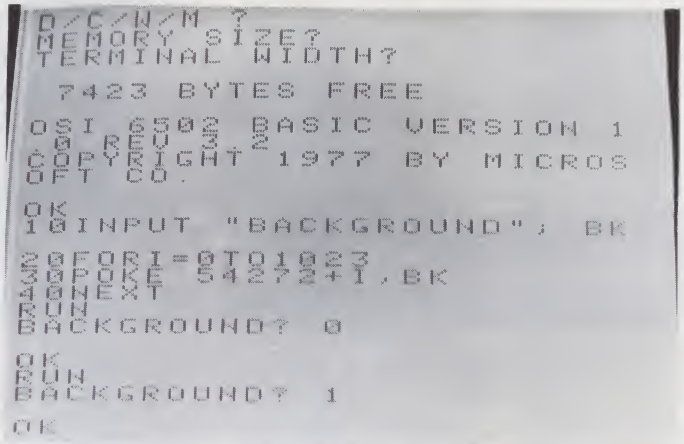


Photo 5c. Result of running Checkout program for "reverse video" display.

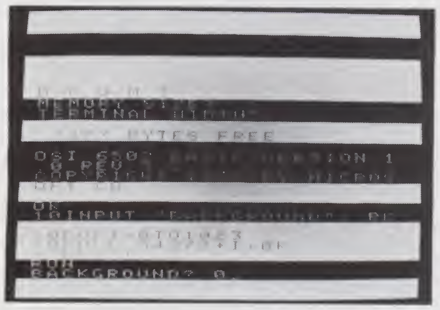


Photo 5a. Checkout program (Listing 1) keyed in for a "normal" display (BK = 0).

late. This results in an incorrectly displayed character. Fortunately, an unused inverter is located at U18, pins 9 (input) and 8 (output). Wire U19, pin 8, to U18, pin 9. Wire U18, pin 8, to U27, pin 3.

Sound engineering practice would require the use of a pull-up resistor to terminate two unused inputs on the 74LS74, and also one on the NAND gate at U56. I found it unnecessary with the circuits used, but if you want, add a 1k, ¼-watt resistor between the unused pin pads at the bottom of U27. Connect the pin pad beside pin 8 to +5. Connect the pin pad beside pin 7 to U27, pins 4 and 1. Also, connect the pin pad beside U27, pin 7, to U56, pin 2.

Photo 3 shows the completed modifications. Except as noted, wire dress is not critical. Several of the wires were placed on the bottom, or wire, side of the circuit board to allow for short, direct runs.

Operation of Modified Video

When you first turn the monitor on, the video display will show the usual random characters, as well as some characters displayed in reverse video. Photos 4a and 4b show this display with two different 21L02 memories at power-up time. Either display is correct. Pressing the break key will clear the characters from the screen but will not alter the black/white pattern of the reverse video. See Photo 4c.

If you have left the toggle switch from the first part of the conversion in the circuit, make sure that it is in the open position. Otherwise, it will be impossible to produce a programmable display.

Bring the system up in BASIC and enter the short program shown in Listing 1 to check out the modification. Run the program, entering 0 when prompted. As the program executes, the screen will be progressively turned black. When the system responds with "OK," execute the program a second time, answering the prompt with "1." Now the screen background will be turned white, thereby displaying any characters in reverse video. Photo 5 (a–c) illustrates this checkout program.

While 0 and 1 were used to indicate normal and reverse video, respectively, any even integer would have produced a normal display; any odd integer would have produced a reverse display. This is because only the least significant digit (D0) of the data bus is stored as a control.

A one-line "program" that will set the video mode is as follows:

FOR I=0 TO 1023:POKE 54272+I,0:NEXT

Executed in command mode—that is, without a line number—this will set the entire screen to either normal or reverse video (poke a 1 for reverse, a 0 for normal). It is especially useful for setting the screen at power-up time.

Programmable reverse video for highlighting text blocks is quite simple: Simply poke into (character display address + 1024) the appropriate code, 0 or 1. As a simple example, Photo 6 shows a single line in reverse video across the bottom of the screen. This is the cursor line in BASIC, where all input is first placed onto the screen. The line's first display location is 54115 (decimal) and is programmed to reverse by poking 54115 + 1024 to 54139 + 1024 with a 1.

The display of a character, or string of characters, can be made dynamically as well. Since relational operators return either a Ø (false) or a 1 (true) value, a program can easily determine when to highlight an area of

```
10 INPUT "BACKGROUND";BK
20 FOR I=Ø TO 1Ø23
30 POKE 54272+I,BK
40 NEXT
```

Listing 1. Reverse Video Checkout program. BK = 0 yields normal video; BK = 1 yields reverse video.

```
-- -------------
-- ER=(BAL<Ø)
-- FOR I=1 TO LEN(STR$(BAL))
-- POKE XXXXX+I,ASC(MID$(BAL),I,1)
-- POKE XXXXX+I+1Ø23,(ER AND 1)
-- -----------
```

Listing 2. Program segment to highlight a character string.

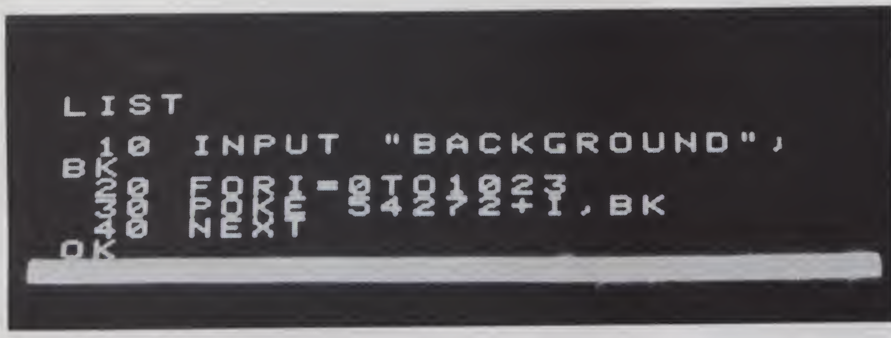


Photo 6. Using reverse video to highlight the cursor line in BASIC.

the display. For example, if your checkbook balance drops below zero, the balance is displayed and highlighted in reverse video. The program segment in Listing 2 illustrates this.

Problems

Two minor problems exist with this implementation of programmable reverse video. While neither is serious, they require mention. In general, they will not affect the utility of the reverse video in operation.

First, you cannot read (peek) the reverse video control memory to determine the state of the control bit for various locations. To add such capability would require another integrated circuit and additional wiring.

The second problem may be a bit more annoying, and exists whenever the screen is programmed into text blocks of reverse video. When the return key is pressed or the screen scrolls, a series of short horizontal bars is flashed onto the screen. These make reading the screen difficult for the instant that they appear, but fortunately exist only for an instant. Since the screen would normally be read while static, this poses no serious problem.

Conclusion

Two methods of providing the OSI C1P with reverse video displays have been detailed. Both methods provide high-quality displays that may be selected without requiring adjustment of the video output circuit or monitor. The first allows only full-screen normal or reverse video. The second permits a point-by-point selection of normal or reverse video of the entire 32 by 32 display screen. Changes and additions to the C1P are kept to a minimum. The total cost of the modifications should not exceed $10.

These approaches to reverse video are easily adaptable to other OSI products, such as the 540 video board. If you try this, remember that pin numbers, circuit locations and signal names will change. ■

References

Lary, Richard A., "Reverse Video from OSI's 540 Board," *Kilobaud Microcomputing* (December 1979).
Ohio Scientific, Inc., *Superboard II Challenger 1P User's Manual*, Ohio Scientific, Inc., 1978.

✓ Reader Service—see page 226

Boot up your system and go directly to the program of your choice. Save time and keystrokes.

For CP/M: Automatic Program Execution on Start-up

By Jon Lindsay

Digital Research's CP/M operating system is a powerful adjuvant to the power of the personal computer. So it was with some excitement that I read an article on "turnkey" CP/M systems in *Creative Computing* (December 1979). But the author's description of an assembly-language program, which made his system operate like the "big" systems by allowing single key program selection on start-up, somehow detracted from another excellent section of the article where he described the technique for making CP/M *automatically* execute a preselected program after reset. I will further describe this technique and give some examples as to how to do it.

With this technique, after you boot up your system (either with a reset or through a monitor and a G + address command), CP/M will sign on in the usual manner and then continue to execute a program of your choice. This program can be anything that you could normally type in and execute in the normal system after the prompt A> is given.

Thus, if you want to have the directory (normally a .COM type command) executed, you may. Or perhaps because of the personnel using your system, you want the machine to come up running BASIC and some selected program such as a MENU, which in itself may allow further BASIC-type program selection. This is now possible.

For an understanding of exactly how CP/M works, see the CP/M reference manuals.

CP/M Modification

When CP/M signs on and gives an A> prompt, any command you type on the screen at that time is entered into a command buffer. The command string is counted and also placed in the same area. The command is executed and the count goes to zero.

After you execute a boot-up of CP/M, but before the A> prompt appears, CP/M looks at the command buffer to see what's there. If the count is zero, it prints the A> prompt and waits for something to be entered. The trick here is to place a command into this area so that after a reset, CP/M will find a command in the command buffer, just waiting for execution. This is relatively easy to accomplish.

During normal CP/M operation, the command buffer area resides in high memory, along with the rest of your CBIOS. However, you need to get to that area as it is being loaded into high memory. Catching it hot off the disk is the best way. Since this area (and the rest of CP/M) is resident on tracks 0 and 1 of a CP/M disk, you must first bring CP/M off the disk and into memory so that it may be saved. Then a DDT operation may be performed on it. Later you will put it all back on the disk.

I assume that you have a correctly running CP/M system. The exact location of the data you are seeking will vary from system to system because of minor modifications to CP/M or bootstrapping necessary to a particular system. I use an ICOM disk system that requires an extra bootstrap program. This throws off the "standard" offset. But as it turns out, there should not be any problem in locating the data area sought. But more about that later.

Cookbooking

First you must get CP/M off the disk and into memory. Do this with the SYSGEN command. The sequence of events is shown in Listing 1.

What you've done is brought CP/M into memory and saved it. Next you must examine it and locate the required command buffer. One of the excellent utilities that comes with CP/M is DDT (Dynamic Debugging Tool). The plan now is to load CP/M back into memory with DDT and operate on it. This is accomplished by typing

A>DDT CPM.COM

```
    type SYSGEN
CP/M responds with:
    SYSGEN VER 1.4
    SOURCE DRIVE NAME (OR RETURN TO SKIP)A (You type the A)
    SOURCE ON A, THEN TYPE RETURN (hit RETURN)
    FUNCTION COMPLETE
    DESTINATION DRIVE NAME (OR RETURN TO REBOOT) (hit RETURN)
Now you will see the prompt A> again. Type
    A>SAVE 32 CPM.COM (Note: if you have version 2.0 of CP/M, use 35 instead of 32)
```

Listing 1.

Jon Lindsay, 3812 N. First St., Fresno, CA 93726.

```
-D0A00
0A00 C3 55 AC C3 51 AC 7F 00 20 20 20 20 20 20 20 20 .U..Q...
0A10 20 20 20 20 20 20 20 20 43 4F 50 59 52 49 47 48          COPYRIGH
0A20 54 20 28 43 29 20 31 39 37 38 2C 20 44 49 47 49 T (C) 1978, DIGI
0A30 54 41 4C 20 52 45 53 45 41 52 43 48 20 20 00 00 TAL RESEARCH  ..
0A40 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0A50 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0A60 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0A70 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0A80 00 00 00 00 00 00 00 00 08 A9 00 00 5F 0E 02 C3 ............_...
0A90 05 00 C5 CD 8C A9 C1 C9 3E 0D CD 8C A9 3E 0A C3 ........>...>..
0AA0 8C A9 C5 CD 98 A9 E1 7E B7 C8 23 E5 CD 8C A9 E1 .......~..#.....
0AB0 C3 A7 A9 0E 0D C3 05 00 5F 0E 0E C3 05 00 0E 0F ........_.......
```

Listing 2.

CP/M responds with

DDT VERS 1.4
NEXT PC
2100 0100

Now, using the commands of DDT, you can locate and change or add data as you choose. Rather than perhaps confuse you with a calculation as to your particular CP/M offset, it is far easier to just look for the area in question. In my system (ICOM), I found it starting at location 0A00H. On a North Star five-inch disk system, I found it at 0900H. In theory, I believe it may be located in the "standard" edition at location 0980H.

But no matter. Look at Listing 2 for the major clue. You are looking for the copyright inscription of Digital Research. It should be roughly in the range suggested above. Once you have found it, the rest are all the same. I might add at this point that you can do this same thing with version 2.0 of CP/M as you can with version 1.4. Using my case as an example, use the D command and type

-D0A00

That which you see in Listing 2 should appear on the screen. In particular, note the line just preceding the copyright start line—(0A00). This is the one that we will operate on. There may be some differences on this line. The underlined portion of the line will vary as to your system size.

0A00 C3 55 AC C3 51 AC 7F 00
20 20 20 20 20 20 20 20 .U..Q...

Now you are ready to enter the hexadecimal values of the command you have selected. For example, if you want to have the directory activated on start-up, you could name it DIR. Therefore, the hex code to be entered would be 44 49 52. Any other .COM file would work as well. It is not necessary to enter ".COM" as code since this is already understood.

In Listing 2, looking at line 0A00, make a note of the 7F at location 0A06. This represents the maximum length of the command buffer. The following byte at location 0A07 is the length of the command string, or the character count. If this byte is zero (00), then CP/M on sign-on will think no command exists and will issue the prompt A>, waiting for a command. While it apparently does not have to be this exact length of the string (any nonzero number will do), it is easy to count the number of bytes of your "command" and insert this number (in hexadecimal equivalent) into this location. Starting at location 0A08, enter the command in hex. The byte immediately after the last byte of your command *must* be a zero (00). Assuming that this is done, you should be able to type the D command again and demonstrate that the

```
-D0A00
0A00 C3 55 AC C3 51 AC 7F 03 44 49 52 00 20 20 20 20 .U..Q...DIR.
0A10 20 20 20 20 20 20 20 20 43 4F 50 59 52 49 47 48          COPYRIGH
0A20 54 20 28 43 29 20 31 39 37 38 2C 20 44 49 47 49 T (C) 1978, DIGI
0A30 54 41 4C 20 52 45 53 45 41 52 43 48 20 20 00 00 TAL RESEARCH  ..
0A40 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0A50 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0A60 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0A70 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0A80 00 00 00 00 00 00 00 00 08 A9 00 00 5F 0E 02 C3 ............_...
0A90 05 00 C5 CD 8C A9 C1 C9 3E 0D CD 8C A9 3E 0A C3 ........>...>..
0AA0 8C A9 C5 CD 98 A9 E1 7E B7 C8 23 E5 CD 8C A9 E1 .......~..#.....
0AB0 C3 A7 A9 0E 0D C3 05 00 5F 0E 0E C3 05 00 0E 0F ........_.......
```

Listing 3.

```
-D0A00
0A00 C3 55 AC C3 51 AC 7F 0B 4D 42 41 53 49 43 20 4D .U..Q...MBASIC M
0A10 45 4E 55 00 20 20 20 20 43 4F 50 59 52 49 47 48 ENU.    COPYRIGH
0A20 54 20 28 43 29 20 31 39 37 38 2C 20 44 49 47 49 T (C) 1978, DIGI
0A30 54 41 4C 20 52 45 53 45 41 52 43 48 20 20 00 00 TAL RESEARCH  ..
0A40 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0A50 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0A60 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0A70 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0A80 00 00 00 00 00 00 00 00 08 A9 00 00 5F 0E 02 C3 ............_...
0A90 05 00 C5 CD 8C A9 C1 C9 3E 0D CD 8C A9 3E 0A C3 ........>....>..
0AA0 8C A9 C5 CD 98 A9 E1 7E B7 C8 23 E5 CD 8C A9 E1 .......~..#.....
0AB0 C3 A7 A9 0E 0D C3 05 00 5F 0E 0E C3 05 00 0E 0F ........_.......
```

Listing 4.

code is correct (see Listing 3).

If you want to use MBASIC (an extended disk BASIC running in CP/M), you could have it execute automatically and then load some particular program that runs under BASIC. In other words, any legitimate command execution after the A> is also possible here. An example of a BASIC program might be something called "MENU." It is set to operate in Listing 4.

So much for the command location. Now you must return this information to the disk resident. Convinced that you have the proper command in place, that the first byte after the 7F is a nonzero byte, and that you have ended the command sequence with a zero (00), you are ready to exit DDT. Type GO. CP/M will give you the A> prompt once again. You may save this by immediately typing

A>SAVE 32 CPM.COM

or you can immediately generate your new CP/M operating system. To do this, type

A>SYSGEN (now hit RETURN)

CP/M responds with

SYSGEN VER 1.4
SOURCE DRIVE NAME (OR RETURN TO SKIP)

Here you hit the return key rather than a drive letter. CP/M responds with

DESTINATION DRIVE NAME (OR RETURN TO REBOOT)A (type A)
DESTINATION ON A, THEN TYPE RETURN (hit RETURN)
FUNCTION COMPLETE
DESTINATION DRIVE NAME (OR RETURN TO REBOOT) (now hit RETURN)

Returning to CP/M command mode (A>), you are ready to give it a try. Hit your reset button to restart your system from scratch. After the sign-on of CP/M and its memory size, it should continue to load your program and execute it. That's proof that you did everything just the way you were supposed to.

If you have trouble, recheck your saved edition of the modified CP/M to see that all of the code has been entered properly. Then try it again. Though I have had no trouble in implementing this technique in several different systems, you might possibly have to redo each step from the beginning to make it right. Its value is that you now have an "automatic" computer. ■

Second Cassette Interface With One IC

By Thomas E. Hutchinson

If you're new to small computers, you probably think one tape recorder is a luxury and two are positively excessive. But some day you'll want to write a program that will read information from one tape, process it and write new data to a second tape.

For example, suppose you have a list of names and addresses stored on a tape as a data file. If someone moves, you'll want to change his address. You can't simply go to that spot on the tape and insert the new information without disturbing nearby data. You have to create a whole new file.

Typically, you'd write a program that would read the data from one tape and write it to the second tape in blocks until the address you want to change is reached. The reading process then stops while you type in the corrected address. This new address is written into the new file on the second tape recorder, and the rest of the names and addresses are alternately read from the first tape recorder and written into the new file on the second.

This method of keeping an address file is only used if the amount of information you wish to store exceeds the 8K or 16K of RAM in your computer. If you have sufficient space in RAM, you would just store your names and addresses in data statements and then save the program in the conventional way on one tape.

Storing Data on Tape

The signal to be recorded on tape comes from pin E on the second cassette port of the PET. This is a square wave of 5-volt amplitude.

I had some success just attenuating this to the proper level for my tape recorder and recording it directly. However, I found that this gives unreliable results with certain tapes. I would save a program in the normal way on the second recorder, and when I attempted to verify that program, I found it was full of errors.

With PEEK(630) for the PET model 2001-8, you can see how many tape errors have been detected during the verify procedure. With some tapes, PEEK(630) was zero every time, and with others, PEEK(630) varied from about 12 upwards even for short programs.

To check whether the tapes themselves were faulty, I repeated the recording and verifying sequence on the tape recorder built into PET. There was no problem here; I got PEEK(630) = 0 every time. Some tapes just do not take kindly to recording square waves from my inexpensive second cassette unit.

I looked through back issues of *Kilobaud* to get ideas from others. In Peter Stark's article "Copying Computer Cassettes" (August 1978, p. 94), the author suggested that his tape recorder did not respond well to square wave input. This appeared related to my difficulties, and following his approach to the problem, I designed an active low-pass filter with a cutoff frequency of 3000 Hz (Fig. 1). The purpose is to round off the corners of the square waves by preventing high-order harmonics of the fundamental frequency of the square wave from passing onto the recording heads.

In Fig. 1 the 5-volt TTL signal enters at the left; it is cut down to 2.5 V by a pair of 10k resistors. It passes through the low-pass filter, which allows frequencies up to 3000 Hz to

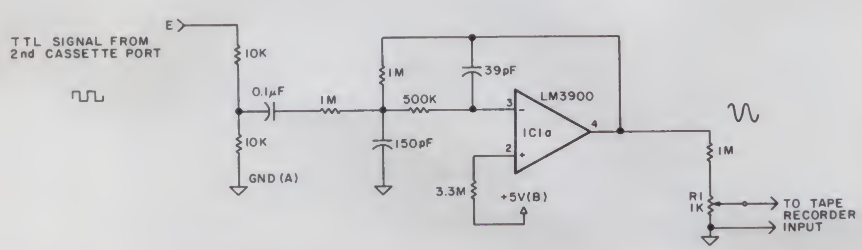


Fig. 1. Low-pass filter rounds off square waves from PET to give better recording.

Thomas E. Hutchinson, 65 Southport St., Apt. 110, Toronto, Ontario, Canada M6S 3N6.

pass with unity gain while attenuating higher frequencies. The output on pin 4 of IC1 is approximately 2.5 volts peak-to-peak, which is cut down to a few millivolts with a resistor network.

The only input on my recorder is a microphone input, which requires only a few millivolts of signal. Some recorders have a high-level auxiliary input, which may accept a signal directly from pin 4 of IC1. The low-pass filter was designed using equations from the *IC Op-Amp Cookbook* by Walter G. Jung.

Reading Data from Tape

The signal from the tape recorder must be squared up and adjusted for 5-volt amplitude before going to the computer. The circuit in Fig. 2 accomplishes this. The signal enters at the left. The level here should be at least 1 volt peak-to-peak. It is clipped by a pair of diodes and then amplified by a factor of 10 by IC1B.

The output on pin 5 is a wave with flat tops and bottoms and an amplitude of 5 volts. To improve the rise and fall times of this wave and make it truly "square," it is fed into the Schmitt trigger constructed around IC1C. The output on pin 9 remains low until the signal from pin 5 rises above 2.6 volts, at which time pin 9 switches high. Pin 9 remains high until the signal from pin 5 falls below 2.1 volts, at which time pin 9 switches low. The sharp corners are thus restored to the square wave coming from the tape. Again, design equations were taken from the *IC Op-Amp Cookbook*.

Turning the Recorder On and Off

A 6-volt signal from pin C on the second cassette interface will drive some tape recorder motors, but not many. As shown in Fig. 3, I use this signal to operate a 6-volt Radio Shack relay, which, in turn, switches the tape recorder on or off by using the remote switch.

The relay is normally open. A manual switch in parallel with the relay contacts allows you to use the fast-forward or rewind buttons when the relay is open. Pin F was designed to go to a switch that would close to ground whenever the play button is depressed on the recorder.

Since I have no such switch, I just grounded pin F as shown in Fig. 2. The only disadvantage with grounding F permanently is that PET will not prompt you with messages such

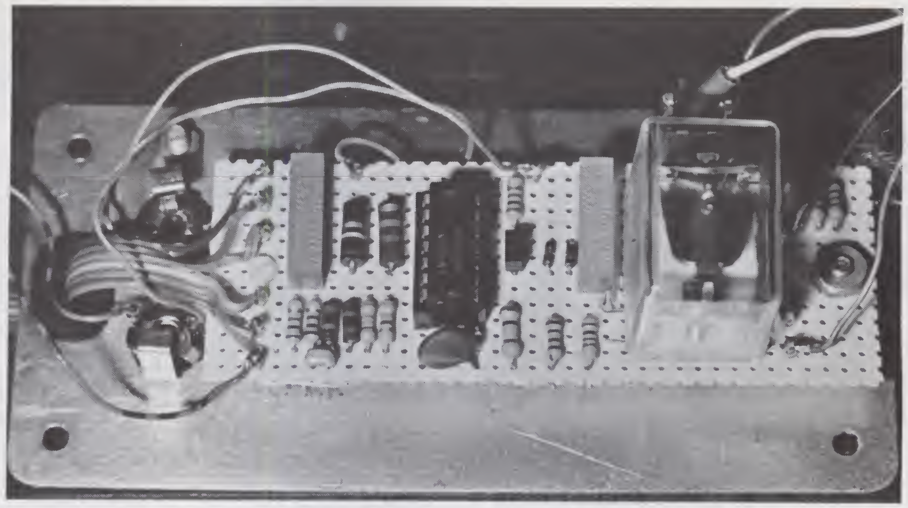


Photo 1. Perfboard is mounted on underside of aluminum lid. Ribbon cable enters at left. Relay is on the right. *(Photos by Dave Lucko)*

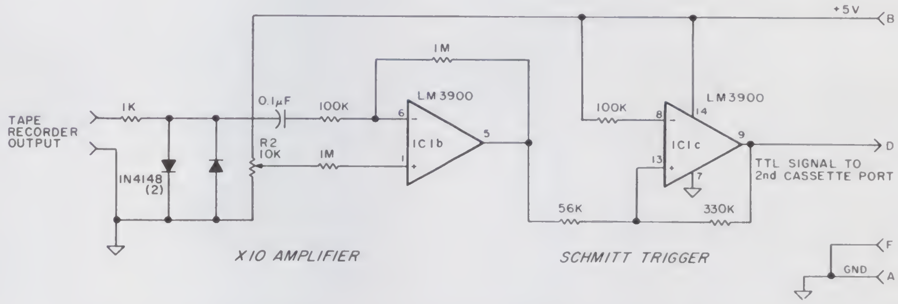


Fig. 2. Tape recorder signal is amplified and changed to a good square wave for PET.

as "press play on cassette 2." Since you obviously have to press the play button when you want to verify, or the play and record buttons when you want to save, I don't find the lack of these messages to be a disadvantage.

Construction

I constructed the entire system on a piece of perfboard and fit it inside a 5×2½ × 1½ inch plastic box from Radio Shack.

One precaution I overlooked cost me a new power supply for my tape recorder. I had three jacks screwed down to a common aluminum panel that formed the lid on the aforementioned plastic box. One jack was for the remote, one was for the "microphone input" and the third was for the "speaker output." Unfortunately, the remote jack was not supposed to be at ground potential as were the speaker and microphone jacks, and I consequently short-circuited and burned out the AC adapter. The cure was to mount the remote jack on the plastic side of the box instead of on the aluminum panel.

I soldered a ribbon cable of six wires to posts on the perfboard. It passes through a rubber grommet. I soldered a six-contact edge connector from Radio Shack to the other end, which plugs into the back of the PET.

One section of IC1 is unused. Its pins may be left floating without harm. R1 and R2 are ten-turn trimpots; since they require an initial adjustment only, they can be mounted inside the box on the circuit board.

Adjustment

The first adjustment is to set R1 in Fig. 1 for the correct recording level. You can conveniently generate a signal for this purpose by first loading a fairly long program (at least one minute of tape time is best) on cassette 1. Then type SAVE"TESTER",2. Press the return key. A signal should appear at the output of IC1A. Press re-

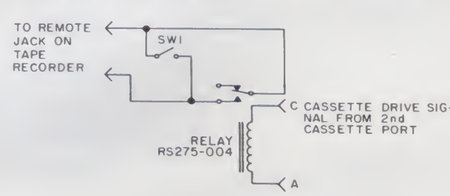


Fig. 3. A relay lets PET turn second cassette motor on and off.

Photo 2. Plastic box houses entire circuit and insulates the remote control jack at bottom right from input and output jacks.

cord and play on cassette 2. Set the record level on the cassette to maximum and adjust R1 so that the signal level meter on your recorder is at the top end of the allowable range, much as you would do if you were recording music.

The second adjustment will be to set R2 in Fig. 2. Disconnect the tape recorder output from IC1B. Connect a dc voltmeter between pin 5 of IC1 and ground. Adjust R2 until the voltmeter reads 2.5 volts. This ensures that the signal output at pin 5 will be a symmetrical square wave when data is fed into IC1B from the recorder.

Everything should now be ready to go. To test it, enter a program from the keyboard or from the first cassette into PET. Type SAVE"TESTER",2. Press the return key. You should hear the relay click. Immediately press play and record on cassette 2. The recorder will stop automatically when the program has been recorded.

Press rewind and turn on SW1 (Fig. 3). After rewinding, turn off SW1. Type VERIFY"TESTER",2. Press the return key. The program should verify normally. Immediately after the verifying has been completed, type PRINT PEEK(630). Press the return key. The answer should be zero if your system has detected no dropout errors. ■

APPLE — JACK

the graphics & games people

New! Super Starbase Gunner

$19.95 DISK

$19.95 DISK

Most shoot-em-up target games are 2-D shoot across the screen type, and quite frankly there is a glut of inferior ones. A need for a new approach exists, such as fast 3-D HIRES simulations with clever and complex challenges. How about shooting into the screen, into 3-D space, where the target is mathematically many feet behind the screen surface? How about computer intelligent targets that shoot back and use strategy and learn? How about all this and the best attributes of the more popular games? Let's include high score, 10 levels of play, snappy sound effects, colorful explosions and real time graphics. Why not go all the way and have a three dimensional gunsight? A real space battle simulation . . . Nah . . . no one would believe it or could even write it. Right?

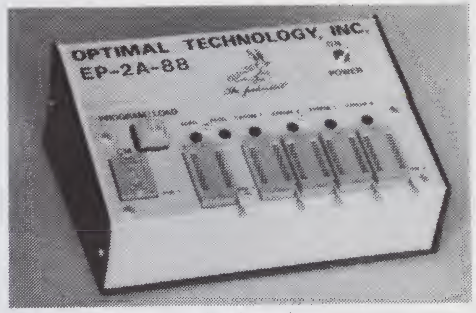
WRONG!! WE HAVE IT . . . and it is SUPER STARBASE GUNNER. We are very excited about this product because it is all the things we wish we had and didn't. And you can have it now with this introductory offer.

SUPER STARBASE GUNNER DISK . . . $19.95 48K with APPLESOFT ROM

AVAILABLE FROM YOUR DEALER OR DIRECT FROM *APPLE—JACK*, BOX 51, CHERRY VALLEY, MA 01611

✓314 (INQUIRIES INVITED)

Model EP-2A-88
EPROM Programmer



Fast as Jackrabbits . . . Well, almost!

In Australia, two rabbits can reproduce over 13 million offspring in 3 years . . . at 105 seconds per 2716, the **EP-2A-88** can reproduce 1,892,160 EPROMS in 3 years. **Single push button control, the EP-2A-88 checks if EPROMS are erased, programs and verifies. It also checks for defective EPROMS.**

Two basic models are available, The EP-2A-88-1 will accept Copy (CM) modules for the 2758, and 2716 EPROMS. The EP-2A-88-2 will accept copy modules for the 2716, 2732 and TMS 2532 EPROMS. Power requirements are 115 VAC 50/60 Hertz at 15 watts.

Part No.	Description	Price
EP-2A-88-1	EPROM Programmer	$490.00
EP-2A-88-2	EPROM Programmer	490.00
CM-50	Copy Module for 2716, TMS 2516 EPROMS	25.00
CM-70	Copy Module for 2758 EPROMS	25.00
CM-20	Copy Module for 2732 EPROMS	25.00
CM-40	Copy Module for TMS-2532 EPROMS	25.00
	Non-Standard Voltage Option (220 v, 240 v, 100 v)	15.00

Optimal Technology, Inc.
Blue Wood 127, Earlysville, Virginia 22936
Phone (804) 973-5482

✓29

SAVE YOUR TIME ...AND YOUR MONEY.

Instant Software has the two best mail list programs available for your TRS-80 Model I and Model II.

Mail/List for Model I and Model II

This mailing list program maintains separate alphabetical and zip code files in *constant sort*. When you add a name to your list, it will *automatically* be inserted into its correct position in the files; therefore it's always ready to print labels!

It will record your information in these fields: NAME, ADDRESS, CITY, STATE, ZIP, PHONE NUMBER, PHONE EXTENSION and a five character CODE field. You have the choice of a 3 line, 4 line, or user defined label format. It can even include (optionally) a message line on your label!

The programs most outstanding feature is its sorting capabilities. Mail/List allows you to choose which names you want to be printed from the whole list. For example, all people in one zip code, or all people named Jones, who are living in a particular city or state. For any name in your list you can assign a code within the CODE field. You can then specify the code when printing labels, and only names with that code will be printed out. You can specify up to 9 different codes!

Every business and organization will save time and money with Mail/list to keep track of customers or members.

TRS-80 Model I version, Order No. 5000RD $99.00.
Requires 16K RAM, Expansion Interface with at least 16K RAM, one disk-drive and a printer.

TRS-80 Model II version, Order No.5001RD $199.00.
Requires 64K RAM and printer.

WRITE FOR OUR NEW CATALOG

TO ORDER: See Your Local Instant Software Dealer or Call Toll-Free 1-800-258-5473.

One-D Mailing List

Here is a mailing list system that can be run on only ONE disk-drive! You can have up to 17 fields of selection for name/address retrieval.

•Disk versatility allows you to add, delete, or change the numerous details stored in the system.

•Features of the One-D Mailing List includes:

•Automatic name sort (alphabetically or by ZIP code). •Rapid access to any name on file. •Easy error correction and recovery. •Prints selective name listings. •Revise or update listings at any time. •Up to 2500 names on-line (with 4 drives). •Prints a list of all names on file. •Prints mailing labels.

This package requires the following minimum system: 1. A TRS-80 Model I Level II microcomputer with 16K RAM. 2. An Expansion interface with 0 to 32K of RAM. 3. A single disk drive (extra drives optional). 4. A printer. 5. Any TRSDOS compatible Disk Operating System.

Order No. 0123RD $24.95.

✓40

Instant Software™

PETERBOROUGH, N.H. 03458

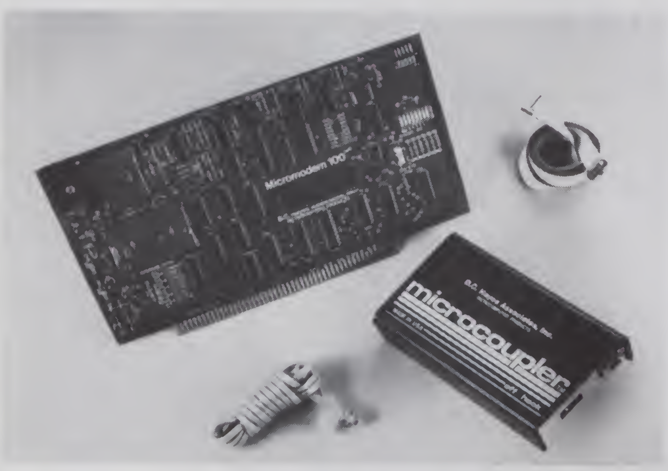
Customer Service
Hayes Microcomputer Products, Inc.
5835 Peachtree Corners East
Norcross, GA 30092

Modem Control

BASIC control program for the Hayes' Micromodem 100 or 80-103A.

Data communications is beginning to receive attention in the small systems field, and articles about it are appearing with greater frequency in many of the popular magazines. This is due to the fact that the telephone network is probably the single most powerful peripheral device that can be connected to a computer. It is also one of the most complex and confusing.

The programming considerations for a telephone line are different from those of the more familiar computer peripherals. First, it is necessary to establish a connection. This means either dialing a number or detecting a ring and answering the phone. Then a carrier signal must be established. The connection must be checked, and the program must be prepared to give up and try again because telephone connections don't always work.



Micromodem 100. (Photo courtesy Hayes Microcomputer Products, Inc.)

Sometimes when you are called, it's not who you expect; and when you dial, you may get a wrong number, the line may be busy or you might get a bad line. Once you get a good connection, it may still go away unexpectedly, and even when it doesn't, you will get some random errors.

This sounds pretty bad, but once you program the computer to worry about these things, the process is mostly automatic, and being able to send data and programs across town or across the country is really worth the trouble.

The Program

In order to make this program as accessible as possible to as many different people as possible, I have written it in a common dialect (Microsoft) of BASIC. This program will run as written under the CP/M version of Microsoft BASIC.

For other BASICs and other versions of Microsoft BASIC, some changes may be required—especially lines 370 and 470 (see below). I have stuck to standard features of the language as much as possible and have generally tried to write the program so that it will be readable, to the extent of sacrificing efficiency where necessary.

The purpose of the program is to perform some special-purpose I/O, so the most important parts of it make heavy use of the IN and OUT features of Microsoft BASIC. This includes keyboard I/O during the communication phase of the program. That is necessary because BASIC otherwise offers no way for a program to test whether a key has been struck or to read a single byte without having to read a whole line.

The only really obscure coding (for which I apologize) is the pair of POKEs at lines 370 and 470. These mysterious POKEs disable part of BASIC's keyboard logic (in the CP/M version) so that the program can take complete control of the keyboard. Between statements, the BASIC interpreter checks the keyboard status to see if a key has been struck. If one has, the intepreter reads it to see if it was a control/C. If it was, it stops running the program and goes into direct mode.

This is essential if the control/C facility is to be able to stop a program that is caught in an infinite loop. It means, however, that a BASIC program that is looping, waiting for a keyboard character, will usually not see any. This happens because between the times the program is looking at the port, the interpreter is also looking, and whenever a character is typed, the interpreter takes it

Program A. Modem Control program.

```
5 REM               80-103A MODEM CONTROL PROGRAM

10 DEFINT A-Z    'MAKE ALL VARIABLES INTEGERS FOR SPEED
20 SKB=&HE4 : DKB=&HE5 : KBB=1 :EOT=4    'KEYBOARD ADDRESSES
30 REM NAME THE MODEM FUNCTIONS
40 M0=128:M1=M0+1:M2=M1+1                'DEVICE ADDRESSES FOR MODEM
50 RI=128:CD=64:OE=16:FE=8:PE=4:TE=2:RRF=1      'STATUS BITS
60 PI=16:SBS=8:LS=2:EPE=1                       'UART CONTROL BITS
70 OH=128:RD=32:ST=16:BK=8:MS=4:TXE=2:BRS=1     'MODEM CONTROL BITS
80 TRUE=-1:FALSE=0:YES$="YES"                   'INSERT SOME ENGLISH
90 REM SET THE DEFAULT OPTIONS
100 B3=FALSE:PARITY=FALSE:EVEN=FALSE:LNG=3
```

More

away, sees (usually) that it's not a control/C and throws it away.

The Mits version of this BASIC has a CONSOLE statement that can be used to turn off the normal console functions. In this case, line 370 should read CONSOLE 0,15, and line 470 should be chosen to restore the correct console function. Remember that the console must be restored to its normal operation when the program is done or its normal functions may be lost.

For Microsoft BASIC Rev. 4.51, lines 370 and 470 should be as in Fig. 1.

Organization of the Program

The first section of Program A is devoted to initialization, much of it for the benefit of the reader. On lines 20-80, I assign names to a variety of useful constants (such as the I/O address of the modem). As far as BASIC will allow, I have made these the same as the names used in the manual.

Line 20 defines the keyboard physical I/O port. SKB is the address of the keyboard status port; DKB is the address of the keyboard data port; and KBB is a mask containing a 1 bit in the bit position of SKB, which indicates the presence of valid data in DKB. This status bit is assumed to be positive true (1 = data present). If this is not the case, then on line 400, the = should be changed to a <>.

Lines 120-160 calibrate the timing loop at line 1040. The loop at 1040 has two exit conditions—when the loop variable Z = T and when the modem status port has a 1 bit corresponding to a 1 bit in the special mask XT. For calibration, the XT mask is set to the transmitter register empty bit, and the loop count T is set to a large value. The modem is then started at 110 baud, its transmitter is filled with zeros and the timing loop is called with a GOSUB. Under these conditions, it will take 100 milliseconds to exit

```
370 POKE &H137B, &HC9
470 POKE &H137B, &HCD
```

Fig. 1.

Program A continued.

```
110 ECHO=FALSE
120 REM CALIBRATE THE TIMING LOOP
130 T=2000:XT=TE                    'PUT LOOP IN CALIBRATE  MODE
140 OUT M2,0:OUT M1,3*LS*EPE              'AT 110 BAUD, 1CHAR TAKES 100 MS.
150 OUT M0,0:OUT M0,0               'FILL THE UART TRANSMIT REGISTERS
160 GOSUB 1040           'RETURNS WHEN ONE HAS BEEN CLOCKED OUT
170 TC=Z:XT=0                       'TC IS # OF ITERATIONS FOR 100 MS.
180 REM NOW ASK A FEW QUESTIONS
190 INPUT "WANT TO CHANGE OPTIONS";I$
200 IF I$<>YE$ GOTO 320
210 INPUT "HIGH BAUD RATE (300)";I$
220 B3=I$=YE$
230 INPUT "GENERATE AND CHECK PARITY";I$
240 PARITY=I$=YE$
250 IF NOT PARITY GOTO 280
260 INPUT "EVEN PARITY";I$
270 EVEN=I$=YE$
280 INPUT "CHARACTER LENGTH IN BITS (NOT INCLUDING PARITY)";I
290 LNG=I-5
300 INPUT "LOCAL KEYBOARD ECHO";I$
310 ECHO=I$=YE$
320 OUT M1,(PI AND NOT PARITY)+(LS*LNG)+(EPE AND EVEN)
330 INPUT "PLACE CALL";I$
340 IF I$=YE$ THEN GOSUB 680 ELSE GOSUB 530
350 IF (INP(M1) AND CD) = 0 GOTO 190     'IF WE DIDN'T CONNECT, GO BACK
360 PRINT "CONNECTION ESTABLISHED"
370 POKE &H1262,&H97                'DISABLE CTRL/C LOGIC IN BASIC!
380 IF (INP(M1) AND CD) = 0 GOTO 450     'CHECK FOR LOST CARRIER
390 IF INP(M1) AND RRF THEN PRINT CHRS(INP(M0));;:GOTO 380 'PRINT INPUT
400 IF (INP(SKB) AND KBB) = 0 GOTO 380   'CHECK KBD
410 IF INP(M1) AND TE THEN C=INP(DKB) AND &H7F : OUT M0,C
420 IF ECHO THEN PRINT CHR$(C);
430 IF C=EOT GOTO 490
440 GOTO 380
450 PRINT "LOST CARRIER, HANGING UP"
460 OUT M2,0              'HANG UP AND TURN OFF MODEM
470 POKE &H1262,&HA7              'PUT CTRL/C BACK
480 GOTO 190
490 PRINT "EOT"
500 PRINT "HANGING UP"
510 GOTO 460
520 REM SUBROUTINE WAITS FOR PHONE TO RING, ANSWERS, AND AWAITS CARRIER
530 PRINT "WAITING FOR PHONE TO RING"
540 IF INP(M1) AND RI GOTO 540  'WAIT FOR RI
550 PRINT "THE PHONE IS RINGING"
560 IF (INP(M1) AND RI) = 0 GOTO 560     'WAIT TIL END OF RING
570 OUT M2,OH+TXE+(B3 AND BRS)           'ANSWER & TURN ON CARRIER
580 PRINT "WAITING FOR CARRIER DETECT"
590 FOR I=1 TO 150                  'GIVE HIM 15 SECS TO RAISE CARRIER
600 IF INP(M1) AND CD THEN RETURN
610 T=TC:GOSUB 1040                 '100 MSEC EACH PASS THRU LOOP
620 NEXT I
630 OUT M2,0                        'HANG UP PHONE
640 PRINT "NO CARRIER -- GIVING UP"
650 RETURN
660 REM END OF PHONE ANSWERING SUBROUTINE
670 REM THIS SUBROUTINE GETS A PHONE #, DIALS IT, AND WAITS FOR CARRIER
680 INPUT "PHONE NUMBER";PH$
690 OUT M2,OH             'TAKE PHONE OFF HOOK
700 T=TC*20:GOSUB 1040  'WAIT 2 SECONDS FOR DIAL TONE
710 PRINT "DIALING - ";
720 FOR I=1 TO LEN(PH$) 'LOOP THRU PHONE NUMBER
730 D=ASC(MIDS(PH$,I,1))                 'GET ONE CHARACTER
740 IF D=42 THEN T=TC*20:GOSUB 1040:NEXT I      '*=2 SEC DELAY
750 D=D-48
760 IF D<0 OR D>9 THEN NEXT I   'IGNORE NON-DIGITS
770 PRINT D;                        'PRINT DIGITS WHILE DIALING
780 TB=TC/2
790 TM=TC/2-4
800 IF D=0 THEN D=10                '0 DIALS AS 10 PULSES
810 FOR J=1 TO D                    'LOOP TO FORM THE PULSES
820 OUT M2,0                        'FORM PULSE BY GOING ON HOOK
830 T=TB:GOSUB 1040      'PULSE LASTS 60 MSEC
840 OUT M2,OH+MS                          'END OF PULSE
850 T=TM:GOSUB 1040      '40 MSEC BETWEEN PULSES
860 NEXT J
870 T=TC*4:GOSUB 1040               '1/2 SEC BETWEEN PULSES
880 NEXT I
890 PRINT
900 PRINT "FINISHED DIALING"
920 PRINT "NOW WAITING FOR HIS CARRIER"
930 FOR I=1 TO 300        'GIVE HIM ABOUT 30 SECONDS
940 T=TC:GOSUB 1040       'WAIT 100 MSEC
942 REM WHEN WE HEAR A CARRIER, WE TURN OURS ON
944 IF (INP(M1) AND CD)=0 GOTO 960
946 OUT M2,OH+MS+TXE+(BRS AND B3)'      TURN ON MODEM
948 X=INP(M0):X=INP(M0)'                CLEAR UART DATA REGISTERS
950 RETURN
960 NEXT I
970 PRINT "NO CARRIER -- GIVING UP"
980 OUT M2,0
990 INPUT "WANT ME TO TRY AGAIN";I$
1000 IF I$=YE$ GOTO 690
1010 RETURN               'COULDN'T DO IT
1020 REM END OF DIALING SUBROUTINE
1030 REM TIMING LOOP -- IT IS CALIBRATED USING MODEM CLOCK
1040 FORZ=1TOT:IFINP(M1)ANDXTTHENRETURNELSENEXTZ
1050 RETURN
```

the loop.

In that time, the loop variable Z will have been incremented a number of times. Its value represents the number of repetitions of the loop needed to equal 100 msec. Later when this timer is used, XT is set to 0, T is set to the desired time in 100 msec units times that value, and a GOSUB is made to 1040.

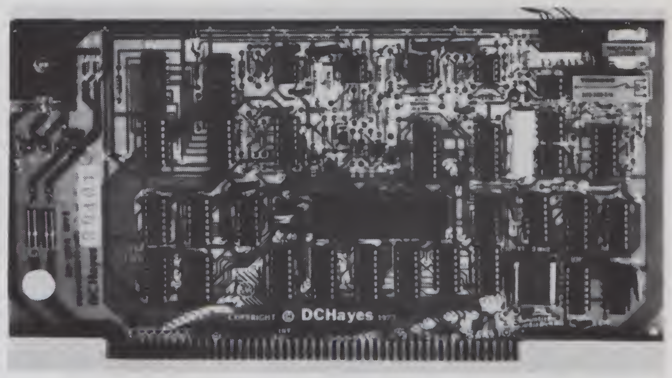
This gives reliable timing with an accuracy of about 10 percent or so on systems with widely differing memory and processor speeds. On my system with a 2 MHz 8080 and no wait states, this loop runs about seven iterations. If your system is much slower, it won't be able to keep up at 300 baud.

Lines 180-330 allow the operator to change the various UART and modem options. This should be fairly clear if you look at the appropriate sections in the modem manual.

Line 340 establishes the telephone connection by calling appropriate subroutines either to answer the phone or to dial a call.

Lines 380-440 are a loop to transfer data between the modem and the display and between the keyboard and the modem. Line 380 checks to make sure the phone connection hasn't gone away. Line 390 checks for received data in the modem and prints it if it finds any. Note that the output device must be at least as fast as the modem (110 or 300 baud, even during carriage returns!), or some data will be lost. Line 400 checks for keyboard input, and line 410 reads it in, strips the parity bit and transmits it if the transmitter is not busy.

If the transmitter is busy, the program will try again the next time through the loop. Line 420 echoes local input if selected, and line 430 checks for EOT (control/D) from the keyboard. This character is used to tell the program that you are done with it and to hang up the phone.



The 80-130A Data Communications Adapter.

Lines 520-650 are a subroutine to wait for the phone to ring. When the phone rings, the subroutine prints a message and waits in line 560 until the end of the ring. Line 570 then turns on the modem. The OH bit takes the modem off the hook (thus answering the phone), the TXE bit turns on the modem carrier and the BRS bit selects the high baud rate if the operator has requested it. It loops (lines 590-620) waiting for a carrier from the other end. The loop exits on a detecting carrier. If no carrier has been detected after 15 seconds, it hangs up again (OUT M2,0) and returns.

Lines 670-1020 are the dialing subroutine. First it gets a phone number in the string PH$ at line 680. Then it takes the phone off the hook at line 690 and waits 2 seconds for the telephone exchange to turn on its dial tone (which indicates its readiness to accept dial pulses). The loop from lines 720-880 loops through the digits of PH$ with I selecting individual digits. Line 730 puts the numeric equivalent of the Ith byte of PH$ into D. If it's an asterisk, line 740 executes a 2-second delay (this is useful on PBX or other systems with a second dial tone).

Lines 750 and 760 make sure the character is a digit and ignore non-digits. This allows you to type in 1(404)555-1212, for example, instead of 14045551212, which is much harder to check. TB and TM are time constants for the break and make times for forming the dial pulses.

The phone company specs these pulses at 61 percent

Program B. File Giver program.

```
5 REM            FILE GIVER PROGRAM

10 DEFINT A-Z   'MAKE ALL VARIABLES INTEGERS FOR SPEED
15 CLEAR 2000
20 REM NAME THE MODEM FUNCTIONS
30 M0=128:M1=M0+1:M2=M1+1                 'DEVICE ADDRESSES FOR MODEM
40 RI=128:CD=64:OE=16:FE=8:PE=4:TE=2:RRF=1       'STATUS BITS
50 PI=16:SBS=8:LS=2:EPE=1                        'UART CONTROL BITS
60 OH=128:RD=32:ST=16:BK=8:MS=4:TXE=2:BRS=1      'MODEM CONTROL BITS
70 TRUE=-1:FALSE=0:YES$="YES"                    'INSERT SOME ENGLISH
80 REM SET THE DEFAULT OPTIONS
90 B3=FALSE:PARITY=TRUE:EVEN=TRUE:LNG=2
100 REM
      CALIBRATE THE TIMING LOOP
110 T=2000:XT=TE                 'PUT LOOP IN CALIBRATE  MODE
120 OUT M2,0:OUT M1,3*LS*EPE              'AT 110 BAUD, 1CHAR TAKES 100 MS.
130 OUT M0,0:OUT M0,0            'FILL THE UART TRANSMIT REGISTERS
140 GOSUB 10520          'RETURNS WHEN ONE HAS BEEN CLOCKED OUT
150 TC=Z:XT=0                    'TC IS # OF ITERATIONS FOR 100 MS.
155 REM
      SET UP TO TRANSMIT FILE
160 OUT M1,(PI AND NOT PARITY)+(LS*LNG)+(EPE AND EVEN)+SBS
1000 ACK=6:NAK=21:CR=13
1100 INPUT "FILE NAME";F$
1200 OPEN "I", #1, F$
1300 GOSUB 10000
1305 IF (INP(M1) AND CD) = 0 GOTO 1300
1310 PRINT "CONNECTION ESTABLISHED"
1312 REM
      GET A LINE FROM THE FILE
1315 DF=FALSE'  CLEAR THE DONE FLAG (WE'RE NOT DONE YET)
1320 IF DF GOTO 1600
1330 IF EOF(1) GOTO 1570
1340 LINE INPUT #1, L$
1345 IF LEN(L$)=0 GOTO 1330
1347 REM
      COMPUTE A CHECKSUM
1350 CS=0
1360 FOR I=1 TO LEN(L$)
1370 CS=CS XOR ASC(MID$(L$,I))
1380 NEXT I
1385 IF CS=ACK THEN CS=0'       MAKE SURE CHECKSUM DOESN'T LOOK LIKE ACK
1387 REM
      SEND THE LINE
1390 FOR I=1 TO LEN(L$) 'LOOP THRU LINE AGAIN
1400 IF INP(M1) AND TE THEN OUT M0,ASC(MID$(L$,I)):GOTO 1460
1405 IF (INP(M1) AND CD) = 0 GOTO 1670
1410 IF (INP(M1) AND RRF) = 0 GOTO 1400
1420 C=INP(M0)
1430 IF C=NAK THEN GOTO 1710
1440 PRINT CHR$(C);
1450 GOTO 1400
1460 NEXT I
1465 REM
      FOLLOW IT WITH A CARRIAGE RETURN
1470 IF (INP(M1) AND TE) = 0 GOTO 1470
```

More

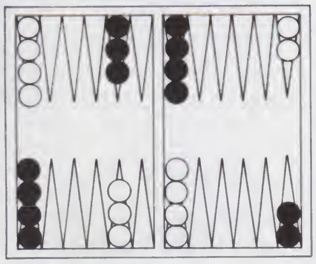
Proven Favorites...
These 10 Popular Games Offer Exciting New Challenges

Here are 10 games that have never lost popularity. They've been played in many forms for many years. And now they're among the favorite software packages in the computerized home-entertainment field.

Don't miss out on the excitement, challenge and downright fun of these programs. Take the word of thousands of satisfied players. They're still tops. Order yours today and play your kind of game.

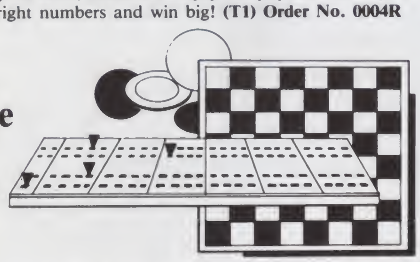
TRS-80*

Beginner's Backgammon and Keno



BEGINNER'S BACKGAMMON/KENO—Why sit alone when you can play these fascinating games: ● BACKGAMMON: Play against the computer in a game that's sure to sharpen your skills; ● KENO: Enjoy this popular Las Vegas gambling game—guess the right numbers and win big! **(T1) Order No. 0004R $7.95.**

Your Cribbage and Checkers Partner



YOUR CRIBBAGE AND CHECKERS PARTNER—● CRIBBAGE is a two-person game that you are sure to enjoy. This is NOT a tutorial—it is a game worthy adversary. ● CHECKERS: An old favorite which follows international rules, including multiple jumps. **(T1) Order No. 0068R $9.95.**

CARDS—A one-player package to let you play, with your computer, these famous games: ● DRAW AND STUD POKER: These programs will keep your game sharp; ● NO-TRUMP BRIDGE: Develop your strategy and (hopefully) increase your skill. **(T1) Order No. 0063R $7.95.**

CHESSMATE-80—This versatile chess opponent gives you a choice of ten levels of play, from the "blitz" level (the computer has 3 seconds to move) to the infinity level (where the computer will consider every possible move—which could take years). This machine-language program is a conservative player and follows all the rules of international play. CHESSMATE-80 can teach you how to move and allow you to set up the board and play end games or special problems. CHESSMATE-80 battled Sargon II to a draw at two minutes a move and beat Microchess 1.5 in six moves. **(T1) Order No. 0057R $19.95.**

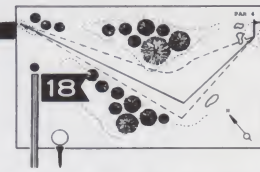
BOWLING—Let your TRS-80 set up the pins and keep score. One player can pick up spares and get strikes. **(T1) Order No. 0033R $7.95.**

OTHELLO—Pit your strategic powers against a merciless, computerized opponent. You play on a board of 64 squares. When you capture your opponent's game disks (by bracketing them with your own disks), they immediately change sides to become members of your set. You can capture several game pieces at one time as long as they are in-line horizontally, vertically or diagonally. Major reversals of score are commonplace as whole blocks of men change side in an instant. **(T1) Order No. 0046R $9.95**

(T1)=TRS-80 Model 1, LevelII, 16K RAM

Apple**

GOLF



GOLF—Without leaving the comfort of your living room you can enjoy a challenging 18 holes of golf with a complete choice of clubs and shooting angles. Now, you need never cancel your game because of rain. One or two players can enjoy this game on your Apple II with Applesoft and 32K. **Order No. 0018A $7.95**

PET***

CASINO I—Gamblers, gather round for: *BLACKJACK—a fun-to-play and tutorial version that allows you to play every combination that they play at the MGM Grand Hotel in Las Vegas; *ROULETTE—also a game *and* a tutorial which could enable you to develop the skill to make a fortune at the tables. **Order No. 0014P $7.95**

CASINO II—This is not simply a craps game, it is a tutorial program that will teach you the odds on every type of bet, so you can steer clear of the sucker bets that impoverish the unwary. Using the exact rules used at the MGM Grand Hotel in Las Vegas, you'll play under actual gambling house conditions. Get the edge you need to "get lucky." **Order No. 0015P $7.95**

CHECKERS & BACCARAT—Using International Rules, you'll play *CHECKERS with your computer and the computer will keep score, time the moves and even tell you when to jump (it'll be so busy, you may have an edge!); *BACCARAT—gives you the choice of playing Las Vegas or Blackjack style and the computer will deal the cards, figure the payoff and keep track of your bankroll. **Order No. 0022P $7.95**

ORDER YOUR INSTANT SOFTWARE TODAY!

Just Call Toll-Free 1-800-258-5473

*A trademark of Tandy Corporation
**A trademark of Apple Computer Inc.
***A trademark of Commodore Business Machines Inc.

WRITE FOR OUR NEW INSTANT SOFTWARE CATALOG

We Guarantee It!

Instant Software Guarantee

OUR PROGRAMS ARE GUARANTEED TO BE QUALITY PRODUCTS. IF NOT COMPLETELY SATISFIED YOU MAY RETURN THE PROGRAM WITHIN 60 DAYS. A CREDIT OR REPLACEMENT WILL BE WILLINGLY GIVEN FOR ANY REASON.

✓40

Instant Software™

PETERBOROUGH, N.H. 03458 603-924-7296

Program B continued.

```
1480 OUT M0,CR
1484 REM
     AND THEN THE CHECKSUM
1490 IF (INP(M1) AND TE) = 0 GOTO 1490
1500 OUT M0,CS                             'SEND CHECKSUM
1502 PRINT L$
1505 IF (INP(M1) AND CD) = 0 GOTO 1670
1510 IF (INP(M1) AND RRF) = 0 GOTO 1505
1520 C=INP(M0) AND 127
1530 IF C=ACK GOTO 1320
1540 IF C=NAK GOTO 1710
1550 PRINT CHR$(C);
1560 GOTO 1510
1570 L$ = "***** END OF FILE *****"
1580 DF=TRUE
1590 GOTO 1350
1600 OUT M2,0                              'HANG UP PHONE
1610 PRINT L$                              'STILL SAYS EOF
1620 CLOSE                                 'LIKE A GOOD BOY
1630 GOTO 1200
1640 IF (INP(M1) AND TE) = 0 GOTO 1640
1650 OUT M0,ACK
1660 GOTO 1350
1670 PRINT "LOST CARRIER -- HANGING UP"
1680 OUT M2,0
1690 CLOSE
1700 GOTO 1200
1710 IF (INP(M1) AND TE)=0 GOTO 1710
1715 T=3*TC:GOSUB 10520
1720 OUT M0,ACK
1730 GOTO 1350
10000 REM SUBROUTINE WAITS FOR PHONE TO RING, ANSWERS, AND AWAITS CARRIER
10010 PRINT "WAITING FOR PHONE TO RING"
10020 IF INP(M1) AND RI GOTO 10020       'WAIT FOR RI
10030 PRINT "THE PHONE IS RINGING"
10040 IF (INP(M1) AND RI) = 0 GOTO 10040         'WAIT TIL END OF RING
10050 OUT M2,OH+TXE+(B3 AND BRS)                 'ANSWER & TURN ON CARRIER
10060 PRINT "WAITING FOR CARRIER DETECT"
10070 FOR I=1 TO 150             'GIVE HIM 15 SECS TO RAISE CARRIER
10080 IF INP(M1) AND CD THEN RETURN
10090 T=TC:GOSUB 10520           '100 MSEC EACH PASS THRU LOOP
10100 NEXT I
10110 OUT M2,0                   'HANG UP PHONE
10120 PRINT "NO CARRIER -- GIVING UP"
10130 RETURN
10140 REM END OF PHONE ANSWERING SUBROUTINE
10510 REM TIMING LOOP -- IT IS CALIBRATED USING MODEM CLOCK
10520 FORZ=1TOT:IFINP(M1)ANDXTTHENRETURNELSENEXTZ
10530 RETURN
```

Program C. File Getter program.

```
5 REM              FILE GETTER PROGRAM

10 DEFINT A-Z   'MAKE ALL VARIABLES INTEGERS FOR SPEED
15 CLEAR 2000          'MAKE ROOM FOR SOME STRINGS
20 REM NAME THE MODEM FUNCTIONS
30 M0=128:M1=M0+1:M2=M1+1                'DEVICE ADDRESSES FOR MODEM
40 RI=128:CD=64:OE=16:FE=8:PE=4:TE=2:RRF=1       'STATUS BITS
50 PI=16:SBS=8:LS=2:EPE=1                        'UART CONTROL BITS
60 OH=128:RD=32:ST=16:BK=8:MS=4:TXE=2:BRS=1      'MODEM CONTROL BITS
70 TRUE=-1:FALSE=0:YE$="YES"                     'INSERT SOME ENGLISH
80 REM SET THE DEFAULT OPTIONS
90 B3=FALSE:PARITY=TRUE:EVEN=TRUE:LNG=2
100 REM
      CALIBRATE THE TIMING LOOP
110 T=2000:XT=TE                'PUT LOOP IN CALIBRATE  MODE
120 OUT M2,0:OUT M1,3*LS*EPE          'AT 110 BAUD, 1CHAR TAKES 100 MS.
130 OUT M0,0:OUT M0,0           'FILL THE UART TRANSMIT REGISTERS
140 GOSUB 10520         'RETURNS WHEN ONE HAS BEEN CLOCKED OUT
150 TC=Z:XT=0                   'TC IS # OF ITERATIONS FOR 100 MS.
155 REM
      SET UP TO GET THE FILE
160 OUT M1,(PI AND NOT PARITY)+(LS*LNG)+(EPE AND EVEN)
900 ACK=6:NAK=21:CR=13  'DEFINE THE NECESSARY CONTROL CHARACTERS
950 EF$="***** END OF FILE *****"        'END OF FILE MESSAGE LINE
1000 ME=OE+PE+FE
1010 PRINT "FILE RECEIVING PROGRAM"
1020 INPUT "FILENAME";F$
1030 OPEN "O",#1,F$
1040 L$=STRING$(150,0)
1050 GOSUB 10150
1060 IF (INP(M1) AND CD)=0 GOTO 1010
1070 PRINT "CONNECTION ESTABLISHED"
1075 REM
      RECEIVE A LINE
1080 I=1
1085 WC=250'    SET UP A TIMEOUT COUNTER
1090 IF INP(M1) AND RRF GOTO 1120
1100 WC=WC-1
1110 IF WC THEN GOTO 1090 ELSE GOTO 1280
1120 IF INP(M1) AND ME GOTO 1310
1130 C=INP(M0) AND 127
1140 IF C=CR GOTO 1175
```

More

break and 39 percent make with a pulse rate of from 8 to 11 pulses per second. In fact, all but the oldest and crankiest of exchanges will take pulses faster than this. If you experiment with dialing rates, keep in mind that you *will* dial some wrong numbers (i.e., don't test your program at 3 AM).

The pulses are created by the loop from lines 810 to 860 with J counting the pulses. Line 870 introduces a 1/2 second delay between digits. The code from line 900 on waits for a carrier (much like the code in the answering routine), except that if it fails, it gives the operator the option of redialing the same number.

General Suggestions

Anytime you are testing new auto-dialing programs, remember that bugs in the program can call wrong numbers. Monitor your program closely when testing and stop it if you know it's dialing a wrong number. Either MASTER RESET or EXTERNAL CLEAR (if your machine has this button) will make the modem hang up immediately (power off will also). On most systems, EXTERNAL CLEAR will have few other effects, and the program will continue to run.

When modifying the data transfer loop at lines 380-440, remember that at 300 baud your program will have to deal with each character received in 33 milliseconds or less. A long BASIC statement can easily take over 10 milliseconds to execute, especially if it must perform floating-point arithmetic. The DEFINT statement at the start of the program is essential. If all the arithmetic were floating-point, this program would have difficulty keeping up at 110 baud, not to mention 300 baud.

Any code that you plan to use here should be timed by putting it in a loop where it will be executed several hundred times and timing it with a watch, etc. First, time the loop without your new code and then with it so that you can subtract the time for executing the loop alone.

LOWEST PRICE - BEST QUALITY

NORTH STAR

North Star Horizon 2

2-5¼ Disk Drives
32K Double Den
Factory assem. & tested
Factory guaranteed
List 3095

only **$2274**

POWERFUL NORTH STAR BASIC FREE
SUPERB FOR BUSINESS & SCIENCE

FACTORY ASSEMBLED & TESTED	LIST	ONLY
HORIZON-1-32K-DOUBLE DEN	$2695	$1994
HORIZON-2-32K-QUAD DENSITY	3595	2674
HORIZON-2-64K-QUAD + HARD DISK	9329	7149
HORIZON RAM ASSM	16K = $389.	32K = $579
HORIZON RAM KIT	SALE! 16K = $314	32K = $469
HORIZON DISK DRIVE SALE DOUB DEN · SAVE!		315
NORTH STAR HARD DISK 18 Mb	4999	3939
PASCAL-PLUS 14,18 OR 36 DIGIT PRECISION		249

SUPERBRAIN

ZENITH

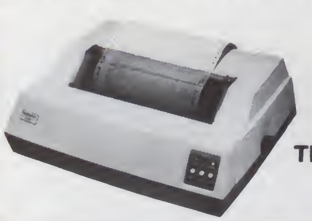
SUPERBRAIN QD 64K
List $3995 only $2995

Z-89 48K
List $2895 only $2299

TERMINALS Z-19 $739

INTERTUBE III only **$725**

DIP-81 PRINTER only **$425**



MICROTEK $675

NEC PRINTER $2699
TRACTOR, THIMBLE, RIBBON

InterSystems

ITHACA INTERSYSTEMS 2A

Z-80A CPU 4 MHz
64K Dynamic RAM
Front panel
V I/O—with interrupts
FDCII Disk Controller
20 slot motherboard

List $3595 You Pay Only **$2995**

PASCAL/Z + THE FASTEST PASCAL $375
GET READY FOR ITHACA'S Z-8000
8086 16 BIT CPU & SUPPORT CARD SEATTLE $575

MORROW 8" DISK

DISCUS 2D + CP/M® 600K ONLY $938
DISCUS 2 + 2 + CP/M® 1.2 MEGA B. $1259
ADD DRIVES 2D = $650 2 + 2 = $975
2D-DUAL + CP/M® + MICROSOFT BASIC = $1555
$2293 VALUE OVER 32% OFF!
WHILE THEY LAST

MORROW HARD DISK
26,000,000 BYTES!!
LIST $4995 ONLY $3995
CP/M® IS INCLUDED!

SAVE ON MEMORY AND PROGRAMS

SYSTEMS MEMORY 64K A & T 4mHz $599
SYSTEMS MEMORY 64K BANK SELECT $789
CENTRAL DATA 64K RAM $599
ITHACA MEMORY 8/16-bit 64K $845
SEATTLE MEMORY 8/16 BIT 16K 4Mhz $356
SSM KITS Z-80 CPU $221
VIDEO BRD V83 4Mhz $412
ANADEX PRINTER DP-9500-1 $1389
CAT NOVATION MODEM $169
TARBELL DISK CONTROLLER DD $445
ECONORAM 2A 8K ASSM $179
NSSE 1-22 & P01 TERRIFIC PROGRAMS ONLY $10. EACH
NORTHWORD 294 MAILMAN 234
INFOMAN $364
TARBELL COMPUTER-PHONE

RCA-COSMAC VP-111 99 RCA-COSMAC VP-711 $199
COLOR! RAINBOW 385
SPECTRUM $289
EZ-80 Machine Language Tutor $25
EZ-CODER Translates English to BASIC $71
ECOSOFT *FULL ACCOUNTING PKG* $315
BOX OF DISKETTES $29
SECRETARY WORD PROCESSOR The Best! $99
TEXTWRITER III Book Writing Program $112
GOFAST NORTH STAR BASIC Speeder Upper $71
Which Computers are BEST? BROCHURE FREE
North Star Documentation refundable w/HRZ $20

AMERICAN SQUARE COMPUTERS BEATS ADV. PRICES

American square Computers

56

919-889-4577 KIVETT DR. JAMESTOWN N.C. 27282 919-883-1105

® CP/M is a registered trademark of Digital Research, Inc.

Program C continued.

```
1150 MID$(L$,I,1)=CHR$(C)
1160 I=I+1
1170 GOTO 1090' WHEW! WHAT A LOT OF STUFF TO GET ONE LITTLE BYTE!
1172 REM
        RECEIVE THE CHECKSUM
1175 WC=250
1180 IF INP(M1) AND RRF  GOTO 1190
1183 WC=WC-1
1187 IF WC THEN GOTO 1180 ELSE GOTO 1310
1190 CS=INP(M0)
1195 REM
        CHECK THE CHECKSUM
1200 FOR  J=1 TO I
1210 CS=CS XOR ASC(MID$(L$,J))
1220 NEXT J
1230 IF CS<>0 AND CS<>ACK GOTO 1310
1235 REM
        NOW WE GOT A LINE, DO SOMETHING WITH IT.
1240 L$=LEFT$(L$,I-1)
1250 PRINT L$
1255 IF L$=EF$ GOTO 1410'       CHECK FOR END OF FILE LINE
1260 PRINT #1,L$
1270 L$=STRING$(150,0)
1272 REM
        NOW LET HIM KNOW WE GOT IT SO HE CAN SEND ANOTHER.
1275 CC=ACK
1280 IF (INP(M1) AND TE)=0 GOTO 1280
1285 IF (INP(M1) AND CD)=0 GOTO 1440
1290 OUT M0,CC
1300 GOTO 1080
1305 REM
        UH-OH, BAD DATA.  SEND HIM NAK, AND GET IT AGAIN.
1310 PRINT "SENDING NAK"
1311 CC=NAK
1315 T=3*TC:GOSUB 10520'        SHORT WAIT LETS GLITCH GO BY
1317 IF (INP(M1) AND CD)=0 GOTO 1440
1320 OUT M0,NAK
1330 WC=250
1340 IF INP(M1) AND RRF GOTO 1370'      WAIT FOR HIM TO ACK OUR NAK.
1350 WC=WC-1
1360 IF WC=0 GOTO 1310
1370 IF INP(M1) AND ME THEN X=INP(M0): GOTO 1340
1380 C=INP(M0) AND 127  'READ CHARACTER AND STRIP PARITY
1390 IF C<> ACK GOTO 1340
1400 GOTO 1080
1405 REM
        WE'RE ALL DONE.
1410 OUT M2,0   'HANG UP PHONE, WE'RE DONE WITH IT
1420 PRINT "FILE SUCCESSFULLY RECEIVED"
1425 CLOSE
1430 GOTO 1010
1440 REM
        HE WENT AWAY.  MIGHT AS WELL HANG IT UP.
1450 OUT M2,0'  HANG IT UP
1460 PRINT "LOST CARRIER -- HANGING UP"
1470 GOTO 1425
10150 REM
        THIS SUBROUTINE GETS A PHONE #, DIALS IT, AND WAITS FOR CARRIER
10160 INPUT "PHONE NUMBER";PH$
10170 OUT M2,OH         'TAKE PHONE OFF HOOK
10180 T=TC*20:GOSUB 10520       'WAIT 2 SECONDS FOR DIAL TONE
10190 PRINT "DIALING - ";
10200 FOR I=1 TO LEN(PH$)       'LOOP THRU PHONE NUMBER
10210 D=ASC(MID$(PH$,I,1))              'GET ONE CHARACTER
10220 IF D=42 THEN T=TC*20:GOSUB 10520:NEXT I   '*=2 SEC DELAY
10230 D=D-48
10240 IF D<0 OR D>9 THEN NEXT I 'IGNORE NON-DIGITS
10250 PRINT D;                  'PRINT DIGITS WHILE DIALING
10260 TB=TC/2
10270 TM=TC/2-4
10280 IF D=0 THEN D=10          '0 DIALS AS 10 PULSES
10290 FOR J=1 TO D              'LOOP TO FORM THE PULSES
10300 OUT M2,0                  'FORM PULSE BY GOING ON HOOK
10310 T=TB:GOSUB 10520  'PULSE LASTS 60 MSEC
10320 OUT M2,OH+MS                      'END OF PULSE
10330 T=TM:GOSUB 10520  '40 MSEC BETWEEN PULSES
10340 NEXT J
10350 T=TC*4:GOSUB 10520                '1/2 SEC BETWEEN PULSES
10360 NEXT I
10370 PRINT
10380 PRINT "FINISHED DIALING"
10400 PRINT "NOW WAITING FOR HIS CARRIER"
10410 FOR I=1 TO 300    'GIVE HIM ABOUT 20 SECONDS
10420 T=TC:GOSUB 10520  'WAIT 100 MSEC
10429 REM WHEN WE HEAR CARRIER, TURN ON OURS AND RETURN.
10430 IF (INP(M1) AND CD)=0 GOTO 10440
10431 OUT M2,OH+MS+TXE+(BRS AND B3)'    TURN ON MODEM
10433 X=INP(M0):X=INP(M0)'              CLEAN OUT GARBAGE IN USART
10435 RETURN
10440 NEXT I
10450 PRINT "NO CARRIER -- GIVING UP"
10460 OUT M2,0
10470 INPUT "WANT ME TO TRY AGAIN";I$
10480 IF I$=YE$ GOTO 10170
10490 RETURN            'COULDN'T DO IT
10500 REM END OF DIALING SUBROUTINE
10510 REM TIMING LOOP -- IT IS CALIBRATED USING MODEM CLOCK
10520 FORZ=1TOT:IFINP(M1)ANDXTTHENRETURNELSENEXTZ
10530 RETURN
```

My timing studies indicate that it should be possible to stuff the characters into an existing string with the MID$ statement, such as MID$(I$,I,1) = CHR$(INP(M0)), but that appending to a new string such as I$ = I$ + CHR$(INP(M0)) is much too slow. It should also be feasible to put the characters into individual elements of an array as integer values.

If you plan to do anything time-consuming with the data you receive, you will have to have some way to make sure that the sending program at the far end will stop sending long enough for you to do it or you'll lose characters.

This can be most easily arranged by having the sending program wait for an acknowledgement at intervals (such as at the end of a line of text). The receiving program can then do what it has to before acknowledging. If you use parity, checksums or other error-checking methods, you may also wish to implement a negative acknowledgement that will request the sending progam to retransmit the data.

Error correction is essential if you plan to transmit such data as object programs, etc., where there will be no human operator to filter out the occasional glitches due to noise and interference on the phone line.

The Giver and Getter programs (Programs B and C) included with this package illustrate a simple implementation of such a protocol. Unfortunately, they are too complex to be run in BASIC at 300 baud. They will run acceptably at 110 baud and should serve as a useful example of the techniques needed to implement an error-checking protocol.

One note of caution: Probably the hardest part of doing a bulletproof protocol is making sure that it won't get hung up by losing data (such as an ACK or an NAK) at some unexpected time. It is generally necessary to have time-outs on everything.

Good luck and have fun. With imagination and a little work, your computer can be on-line via microwaves and satellites to most of the world. ■

Instant Software makes them shine.

Finishing School For Programmers

By David C. Goodfellow

For centuries, young men and women have been sent to private schools to be honed and polished. Computer owners now have something similar for their programs—Instant Software.

This school doesn't polish the program itself—it polishes the programmer by showing him where to apply the polish.

A year ago I sent a Client File/Billing program to ISI for possible marketing. They agreed that the program had merit, but asked me to make a few improvements. Their letter included criticisms from three independent associate editors.

The criticisms pointed to 23 items that needed work. This was a surprise. I didn't think the program was perfect, but I didn't realize it was complex enough to even have 23 elements.

I began at once to make improvements, only to find that a complete rewrite would be easier. Even though I changed little, the rewrite took me around 100 hours. The reason was simple: I didn't know how to do many of the things requested. I spent much of the time educating myself.

Programming Tips

How to round off dollars and cents accurately. The routine I had been using added a penny when no round-off was needed. To avoid this, I had structured my bills so that round-off was always necessary. Now the program does it right, with no hedging on my part.

How to line up dollar amounts on the decimal point. I had been letting the chips fall where they may, and they looked more like buffalo chips than anything else.

How to position number columns exactly. The pre-ISI version positioned columns by counting spaces to the right of a tab. Trouble was, varying lengths of material to the left of the tab could change the tab position, leaving me in left field. The new version positions from the right edge of the paper and (to a point) doesn't care about the length of the preceding material on that line.

This school polishes the programmer by showing him where to apply the polish.

How to delete a record from a disk file and free up the space allotted it. The pre-ISI version simply filled the record with nulls. This worked, at least to the extent that the unwanted data were gone. But it printed a lot of empty lines in hard copy, slowed down searches by requiring the computer to search empty records and sprinkled the disk with unusable records.

The new version rewrites the file, renumbering the valid records, and throws away the empties.

How to spruce up the display, speed up the operation and communicate with the user. The program had always been easy for me to use, because by the time I had finished writing it I was thoroughly familiar with its operation. It hadn't occurred to me that strangers would have more of a problem. The ISI editors let me have it for that, and I quickly learned what "user-oriented" was all about.

Lesson Learned

Client/File Billing (version two) went back to ISI last week, physically fit, fingernails cleaned, hair combed and shoes shining. Royalty checks, fame and fortune fill my dreams. But even if the program never earns me a nickel, the experience has been invaluable. It has given me a program exactly suited to my own needs and has made a better programmer of me.

ISI didn't teach me any programming tricks. It just gave me the incentive to learn and to make the program approach its potential excellence.

ISI is better than prep schools for children. This school polishes the father, allowing him to polish the child, and costs nothing but the time and effort required for any worthwhile project. ■

David C. Goodfellow, 13026 13th S.W., Seattle, WA 98146.

CALENDAR

AMUS Convention

The first annual Alpha Micro User's Society Convention will be held January 17 through 23 at the Deauville Hotel in Miami Beach, FL. The convention will feature seminars; conferences; demonstrations; and meetings for businessmen, programmers and analysts based on the Alpha Micro system. For further information, contact William L. Miller & Associates, 8389 S.W. 151 St., Miami, FL 33158.

Othello Tournament

An Othello tournament open to individuals or teams will be held at the University of California at Santa Cruz on January 17 and 18. To register, send your name(s), program designation and equipment description by Jan. 10 to Professor Peter W. Frey, 421 Kerr Hall, University of California, Santa Cruz, CA 95064.

Educational Software Symposium

An educational software symposium will be conducted on January 17 and 18 at the Holiday Inn in Bridgeport, CT. Seminars by leaders in the educational software field, as well as exhibits and discussions, will be featured. Contact Monica Kantrowitz, president, Queue, 5 Chapel Hill Drive, Fairfield, CT 06432.

Call for Speakers/Session Leaders

A call for speakers and conference session leaders has been issued for the 6th West Coast Computer Faire to be held in San Francisco's Civic Auditorium, April 3-5, 1981. Those interested can obtain further details from Computer Faire, 333 Swett Road, Woodside, CA 94062.

CORRECTIONS

Robin B. Moore has forwarded the listing changes for the Editor and Demo programs that he neglected to include with the article ("Graphics Character Generator," August 1980, p. 106). See Listing 1.

He also notes that if Listing 2 in the article is entered using the Apple's mini-assembler, it will not assemble as shown. This is due to a bug in the Microproducts assembler that he used. To avoid the problem, substitute the following two lines in Listing 2:

```
083E-8D   1E   10   STA $101E
0841-AD   1F   10   LDA $101F
```

Then enter the Apple's Monitor program and patch as shown below:

```
* 840: 0 (carriage return)
*843: 0 (carriage return)
```

Save the program

One zero can make quite a difference, as Merrill Lessley ("Build a Computer Controlled Triac Dimmer," August) points out. The end of the second paragraph in his article (p. 92) should read "2000 watt load," not "200."

When saving a program on the PET, the "ending address" that must be inserted into page zero must be the address *after* the last address you wish to save. As described in "PET Mini Monitor" (October), the last byte of data will not be saved. The following correction should be substituted in Table 3 on p. 89, as well as on p. 91:

```
00E5 XX 3B enter ending address Lo byte plus 01
```

Maybe a better way of stating "ending address" would be "last address plus one."

```
DEMO PROGRAM
  ADD:     507  X0=Y0=COLR: INIT=-12288: POSN=-11527: LINE=-11500: PLOT=-
                11506
  CHANGE:  150  Y0=9: X0=7: CALL POSN: X0=273: CALL LINE
           240  X0=0: Y0=34: CALL POSN: X0=69: CALL LINE
           250  Y0=121: CALL POSN: X0=0: CALL LINE
           380  FOR J=1 TO 20: Y0=110+Y(3): X0=138+X(3): CALL POSN
           390  FOR I=0 TO 3: Y0=110+Y(I): X0=138+X(I): CALL LINE
           520  LINK=2048: HOME=-936
           540  POKE TL,0: POKE TH,9: COLR=255
EDITOR PROGRAM
  CHANGE:  430  CALL INIT: COLR=255: FOR I=0 TO 8: X0=3: Y0=3+16*I: CALL POSN
           440  X0=171: CALL LINE: NEXT I: Y0=157: X0=3: CALL POSN: X0=171: CALL
                LINE
           450  FOR I=0 TO 8: X0=3+14*I: Y0=3: CALL POSN: Y0=131: CALL LINE: NEXT I
           460  X0=171: Y0=3: CALL POSN: Y0=157: CALL LINE
           470  X0=115: CALL POSN: Y0=131: CALL LINE: X0=3: CALL POSN: Y0=157:
                CALL LINE
           570  Y0=3: X0=196: CALL POSN: X0=257: CALL LINE: Y0=77: CALL LINE
           580  X0=196: CALL LINE: Y0=3: CALL LINE
           590  Y0=5: X0=199: CALL POSN: X0=254: CALL LINE: Y0=75: CALL LINE
           600  X0=199: CALL LINE: Y0=5: CALL LINE: RETURN
          1660  X0=Y0=COLR=I=J=K: PAGE=806: TBLE=2304: DIM K$(40),N$(26): D$="":
                REM CNTRL-D
          1680  INIT=-12288: LINK=2048: HOME=-936: LINE=-11500: POSN=-11527:
                PLOT=-11506
```

Listing 1. Changes to CHAR-GRAF for Programmer's Aid #1 ROM.

DEALER DIRECTORY

Phoenix, AZ

Most complete electronics store in Arizona. Books, magazines, prototyping equipment, building supplies & components of all kinds. Free Catalog. Open six days a week. **Tri-Tek Inc., 7808 N. 27th Ave., Phoenix, AZ 85021.**

El Monte, CA

Ohio Scientific specialist in the San Gabriel valley serving greater Los Angeles. Full product line on display. Specializing in business computers. In-house service. Custom programming. Terminals. Printers. Open Mon–Sat, 9 AM–7 PM. **Computer & Video, 3380 Flair Dr., Suite 207, El Monte, CA 91731, 572-7292.**

San Jose, CA

Bay area's newest computer store. Featuring the new Texas Instruments TI 99/4 home & business computer. Software for TRS-80, Apple, PET, etc. Magazines. **Hobbi-Tronics, 1378 S. Bascom Ave., San Jose, CA 95128, 998-1103.**

San Jose, CA

New and used computer products—specializing in S-100 boards, printers, drives, chassis and complete systems, as well as supplies and parts—Imsai, Tandon, Diablo—5000 sq. ft. **W/W Component Supply, Inc., 1771 Junction Ave., San Jose, CA 95112.**

Denver, CO

Tantalizing software for your microcomputer. We have software applications for both home and business. Ask for prices on diskettes, tapes, custom forms, checks, etc. **The Software Gourmet, 1111 South Pearl St., Denver, CO 80210, 777-8890.**

Sarasota, FL

Dynabyte computer systems, Hazeltine and NEC, Word-Star, Structured Systems accounting. Consulting, training, sales, service. **Glisco, Inc., 4001 Roberts Point Rd., Sarasota, FL 33581, 349-0200.**

Sarasota, FL

Real-Time/Industrial Software Engineering Consultants. Top-down design and structured techniques utilized. **Advanced Computer Concepts, Inc., PO Box 15710, Sarasota, FL 33579.**

Tampa, FL

Apple Computer sales and service. S-100 boards from SSM, Godbout, Thinker Toys, California Computer Systems. Computer books and magazines. **AMF Microcomputer Center, Inc., 11158 N. 30th Street, Tampa, FL 33612, 971-4072, 977-0708.**

Aurora, IL

Microcomputer systems for home or business; peripherals, software, books & magazines. Apple, Hewlett-Packard, North Star, Cromemco systems. IDS-440G printer w/Apple graphics, New HP-85 & HP calculators. **Farnsworth Computer Center, 1891 N. Farnsworth Ave., Aurora, IL 60505, 851-3888.**

Chicago, IL

Computer Hardware/Specialists for home and business. Largest selection of computer books, magazines and copyrighted software in Illinois. Experienced factory-trained service department. Feature Apple, Alpha Microsystems and Hewlett-Packard calculators and accessories. **Data Domain of Schaumburg, 1612 E. Algonquin Rd., Schaumburg, IL 60195, 397-8700.**

Chicago, IL

Brand new lowest prices, never undersold, postpaid in USA—Teletype 43 keyboard printers, Okidata & Integral Data printers, SS-50 bus computers, peripherals & business software. **Data Mart, 914 East Waverly Street, Arlington Heights, IL 60004, 398-8525.**

Garden City, MI

Books, Magazines, Hardware, and Software for Apple, Northstar, TRS-80, and PET. **Computer Center, 28251 Ford Rd., Garden City, MI 48135, 425-2470.**

Hannibal, MO

Ohio Scientific products, modifications, service, software. 8" disk for C1p, C4p. Process control specialist. **E&I Technical Service, 5300 Paris Gravel Road, Hannibal, MO 63401, 248-0084.**

Kinston, NC

"Rent-All" CP/M software for multiple rental locations accommodates 7000+ items. Shows profitability by item/store; moves items between stores by demand . . . more. Dealer inquiries invited. **Southern Digital Systems, Inc., Suite 806A, Vernon Park Mall, Kinston, NC 28501, 527-4691.**

Portland, OR

Ohio Scientific specialists for business and personal computers. Local service. Terminals, printers, custom programming. Full OSI product line on display! 10 AM to 6 PM M–F. **Fial Computer, 11266 SE 21st Ave., Milwaukie, OR 97222, 654-9574.**

Milwaukee, WI

Specializing in the CBM-PET, business, personal, educational, industrial, telecomputing systems. Consulting, modems, printers, books, accessories, magazines, supplies, peripherals, timesharing. Factory authorized service. Convenient freeway access. **PETTED micro systems, 4265 W. Loomis Rd., Milwaukee, WI 53221, 282-4181.**

Call or visit the dealer nearest you!

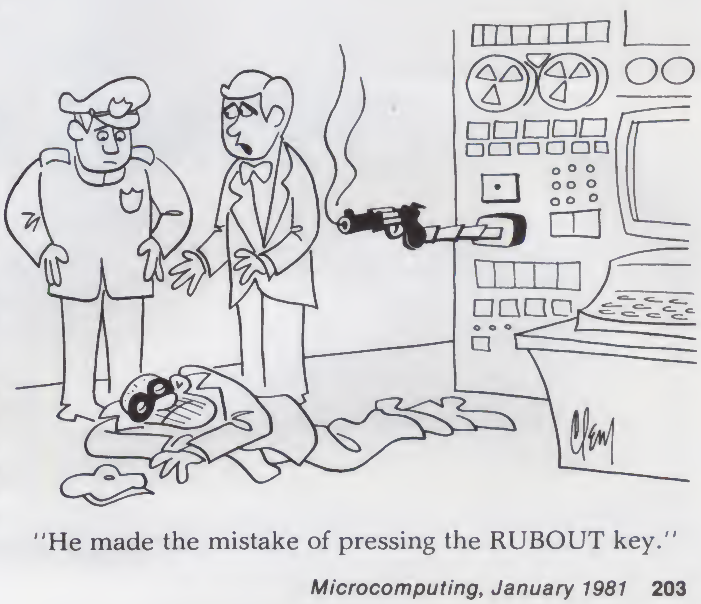
Dealers: Listings are **$15** per month in prepaid quarterly payments, or one yearly payment of **$150**, also prepaid. Ads include 25 words describing your products and services plus your company name, address and phone. (No area codes or merchandise prices, please.) Call Marcia at 603-924-7138 or write *Kilobaud MICROCOMPUTING*, Ad Department, Peterborough NH 03458.

NEXT MONTH:

Kilobaud Microcomputing travels around the world next month as it looks at business, educational and hobbyist applications in foreign countries. Included among the topics to be examined in this international issue are:

- Computers revolutionizing Egypt
- Microcomputing industries flocking to Ireland
- Government support of private microcomputing companies in India
- Spreading the good word of microcomputing in Indonesia and Puerto Rico
- An Englishman's view of Silicon Valley
- Videotext systems in England and the U.S.
- The popularity of the PET in France
- Computerists in Argentina and Australia coping with high prices and the lack of availability of microcomputing parts

You'll also be entertained as Publisher Wayne Green recounts his most recent experiences in the Far East and gives his impressions of the latest developments in microcomputing from that part of the world.



"He made the mistake of pressing the RUBOUT key."

MICRO QUIZ

(from page 23)

Answer: 8.

S is incremented whenever N is divisible by index variable D. Thus, S equals the number of factors of N, excluding 1 and N.

CLASSIFIEDS

Classified advertisements are intended for use by persons desiring to buy, sell or trade used computer equipment. No commercial ads are accepted.

Two sizes of ads are available. The $5 box allows up to 5 lines of about 35 characters per line, including spaces and punctuation. The $10 box allows up to 10 lines. Minimize use of capital letters to save space. No special layouts allowed. Payment is required in advance with ad copy. We cannot bill or accept credit.

Advertising text and payment must reach us 60 days in advance of publication (i.e., copy for March issue, mailed in February, must be here by Jan. 1). The publisher reserves the right to refuse questionable or inapplicable advertisements. Mail copy with payment to: **Classifieds, Kilobaud Microcomputing, Peterborough NH 03458**. Do not include any other material with your ad as it may be delayed.

For Sale: Kim-1 with power supply 5 & 12 V, $125. Kimsi S-100, $75. 2 Mem-1 8K RAM by WMC, $75 ea. Ithaca Audio: EROM 16K with 12-2708s, $70: I-A video display bd, 64 × 16 char, $50. 64 2102 RAM chips, $.50 each. Call Steve at 612-459-0533 after 6 PM.

Sol Terminal Computer, word processor, mailing list, BASIC capab., including monitor, two disk drives, IBM printer, 20 diskettes and all manuals. Top cond., value 7K offered for $4000 or best offer. 213-372-9937, Robert Francois.

Centronics 704 180 cps bi-dir (demo), $1,495 (u-ship). Also new BASF mini-floppy drive (Shugart comp.), $150. 4116s 8/$32. SD Sales Expandoram I (used), $75 V.M. Faulkner, RR 2, Box 294A, Yorktown, IN 47396, 317-289-4138.

Two EXECuport portable terminals 10–30 cps. 80 col. printer modem. Both parallel and serial IO ports for use as printer. See ad. pg. 93, July 80 KB for details. Both units good working cond. Will ship UPS COD, no risk, pay on delivery. Model 300—$375, 320—$475. Call Warren (no coll.) at 509-534-8088, eves. Spokane, WA.

For Sale: SWTPC CT-64 terminal, 16 × 64 × 2 & features. Needs monitor. $225 or best. Also 2400-baud cassette I/O board & software, $20. William Lee, 337 E. Naples, Chula Vista, CA 92011, 714-697-3176.

Tel-It message and inventory computer system. Z-80 based terminal with 16 char. readout (no crt). Built-in tape drive, real-time clock. Will interface to printer. Cost $900, asking $500; call Bob Loveless at 714-689-7800.

For Sale: AJ 841 IO printer-terminal. IBM Selectric mech. 130 ch/line. New. Plus Apple parallel printer interface card. $750. Virginia Stern, 215 E. 11 St., NY, NY 10003. 212-477-6634.

Use the Classifieds

For Sale: All issues of Kilobaud Microcomputing. Issue #1 through December 1980. COD postage collect to best bid. Ed C. Epp, Goshen College, Goshen, IN 46526.

SWTPC 6800 for sale: 20K, serial interface, AC-30, good condition. Sell for best offer over $300. MM Computer Club, 333 75th Street, Downers Grove, IL 60516.

For Sale: TRS-80. 16K L II with numeric keypad. $550. Heath H14 printer 110–9600 baud. Adjustable pin feed. TRS-232 interface for connection to TRS-80 without expansion interface. Both for $550. Call or write Tony Greaves, 1370 Niagara Falls Boulevard, Tonawanda, NY 14150. 716-838-4957.

Wanted: used cassette software for Apple II & books on computing. Send list for my offer. Ray Gabriel, PO Box 175, Madison, OH 44057.

TRS-80 Model I 48K with exp interface keypad cassette recorder covers software Selectric printer & more. 2 disk drives, mint condition. 25–30% off. R. Clore, Jackson, MI, 517-782-4101. After 4PM 563-2978.

For Sale: Working TC-8 Hi-Speed tape system, $90. Gone Disk. Inquiries answered. G. Clark, 32 High St., Newport, ME 04953. After 6, 207-368-5295.

For Sale: New SA-400 mini-disk, $275. FD-250 quad-density drives for North Star, $325. Micromation dual-density controller, dual 8" drives, with CP/M and many extras. Excellent condition. Make offer. Working S-100 boards as follows: Serial I/O, $48; Parallel I/O, $48; Polymorphic Video, $99; Vector-Graphic ROM/RAM w/1K RAM, $48; Hardware interface S-100 to Flex-O-Writer Printer, $48; Cherry-Microswitch Keyboard w/enclosure, $48; Altair with 16-slot motherboard, heavy duty Vector power supply, custom front panel and 8080 CPU (really nice mainframe!), $299. Fred Manthey, Box 619, Mullan, ID 83846, 208-744-1328.

MOVING?

Let us know 8 weeks in advance so that you won't miss a single issue of *Kilobaud Microcomputing*. Attach old label where indicated and print new address in space provided. Also include your mailing label whenever you write concerning your subscription. It helps us serve you promptly.

☐ Address change only
☐ Extend subscription
☐ Enter new subscription
☐ 1 year $18.00
☐ Payment enclosed (1 extra BONUS issue)
☐ Bill me later

AFFIX LABEL

If you have no label handy, print OLD address here.

Name ____ Call ____
Address ____
City ____ State ____ Zip ____

print NEW address here:

Name ____ Call ____
Address ____
City ____ State ____ Zip ____

Kilobaud **Microcomputing** P.O. Box 997 • Farmingdale NY 11737

Dear Subscriber:

Kilobaud Microcomputing does not keep subscription records on the premises. Therefore, calling the Peterborough offices doesn't solve your subscription problem.

To quickly solve your problem, please send your most recent address label and a description of the problem to:

Kilobaud Microcomputing
Subscription Department
PO Box 997
Farmingdale, NY 11737

Please allow the subscription department at least two weeks for an answer or a solution to your problem.

Thank you and enjoy your subscription.

Sincerely,

Debra L. Boudrieau
Circulation Manager

kb microcomputing book nook

3 NEW BOOKS

● **SOME OF THE BEST FROM KILOBAUD/MICROCOMPUTING**—BK7311—A collection of the best articles that have recently appeared in Kilobaud/MICROCOMPUTING. Included is material on the TRS-80 and PET systems, CP/M, the 8080/8085/Z80 chips, the ASR-33 terminal. Data base management, word processing, text editors and file structures are covered too. Programming techniques and hardcore hardware construction projects for modems, high speed cassette interfaces and TVTs are also included in this large format, 200 plus page edition. $10.95.*

● **UNDERSTANDING AND PROGRAMMING MICROCOMPUTERS**—BK7382—A valuable addition to your computing library. This two part text includes the best articles that have appeared in 73 and Kilobaud Microcomputing magazines on the hardware and software aspects of the new microcomputing hobby. Well known authors and well structured text helps the reader get involved in America's fastest growing hobby. $10.95*

● **40 COMPUTER GAMES**—BK7381—Forty games in all in nine different categories. Games for large and small systems, and even a section on calculator games. Many versions of BASIC used and a wide variety of systems represented. A must for the serious computer gamesman. $7.95*

INTRODUCTORY

● **HOBBY COMPUTERS ARE HERE!**—BK7322—If you (or a friend) want to come up to speed on how computers work . . . hardware and software . . . this is an excellent book. It starts with the fundamentals and explains the circuits, and the basics of programming. This book has the highest recommendations as a teaching aid for newcomers. $4.95.*

● **THE NEW HOBBY COMPUTERS**—BK7340—This book takes it from where "HOBBY COMPUTERS ARE HERE!" leaves off, with chapters on Large Scale Integration, how to choose a microprocessor chip, an introduction to programming, low cost I/O for a computer, computer arithmetic, checking memory boards . . . and much, much more! Don't miss this tremendous value! Only $4.95.*

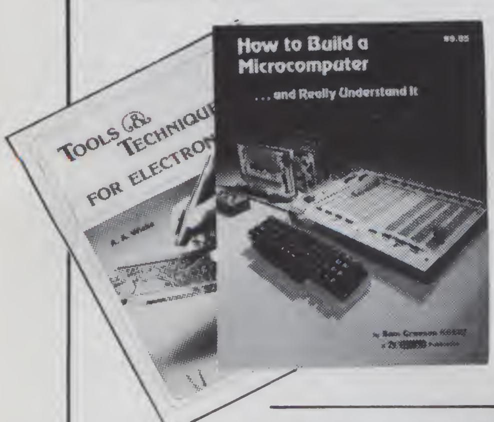
INTRODUCTION TO MICROCOMPUTERS (VOL. 0→III)

● **AN INTRODUCTION TO MICROCOMPUTERS, VOL. 0**—BK1130—The Beginner's Book—Written for readers who know nothing about computers—for those who have an interest in how to use computers—and for everyone else who must live with computers and should know a little about them. The first in a series of 4 volumes, this book will explain how computers work and what they can do. Computers have become an integral part of life and society. During any given day you are affected by computers, so start learning more about them with Volume 0. $7.95.*

● **VOL. I**—BK1030—**2nd Edition completely revised.** Dedicated to the basic concepts of microcomputers and hardware theory. The purpose of Volume I is to give you a thorough understanding of what microcomputers are. From basic concepts (which are covered in detail), Volume I builds the necessary components of a microcomputer system. This book highlights the difference between minicomputers and microcomputers. $12.99.*

● **VOL. II**—BK1040 (with binder)—Contains descriptions of individual microprocessors and support devices used only with the parent microprocessor. Volume II describes all available chips. $31.99*

● **VOL. III**—BK1133 (with binder)—Contains descriptions of all support devices that can be used with any microprocessor. $21.99*



● **HOW TO BUILD A MICROCOMPUTER—AND REALLY UNDERSTAND IT**—BK7325—by Sam Creason. The electronics hobbyist who wants to build his own microcomputer system now has a practical "How-To" guidebook. This book is a combination technical manual and programming guide that takes the hobbyist step-by-step through the design, construction, testing and debugging of a complete microcomputer system. Must reading for anyone desiring a true understanding of small computer systems. $9.95.*

● **TOOLS & TECHNIQUES FOR ELECTRONICS**—BK7348—by A. A. Wicks is an easy-to-understand book written for the beginning kit builder as well as the experienced hobbyist. It has numerous pictures and descriptions of the safe and correct ways to use basic and specialized tools for electronic projects as well as specialized metal working tools and the chemical aids which are used in repair shops. $4.95.*

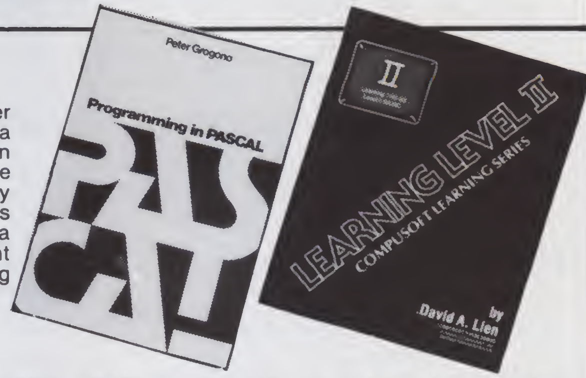
*Use the order card in the back of this magazine or itemize your order on a separate piece of paper and mail to Kilobaud *Microcomputing* Book Department • Peterborough NH 03458. Be sure to include check or detailed credit card information. No C.O.D. orders accepted. All orders add $1.00 handling. Please allow 4-6 weeks for delivery. Questions regarding your order? Please write to Customer Service at the above address.

"PRICES SUBJECT TO CHANGE WITHOUT NOTICE"

FOR TOLL FREE ORDERING CALL 1-800-258-5473

kb microcomputing book nook

BASIC AND PASCAL

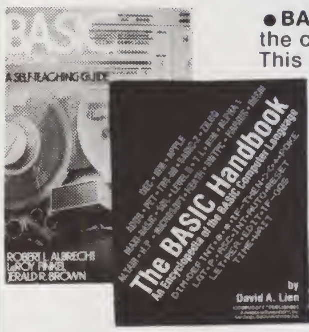


NEW REVISED EDITION

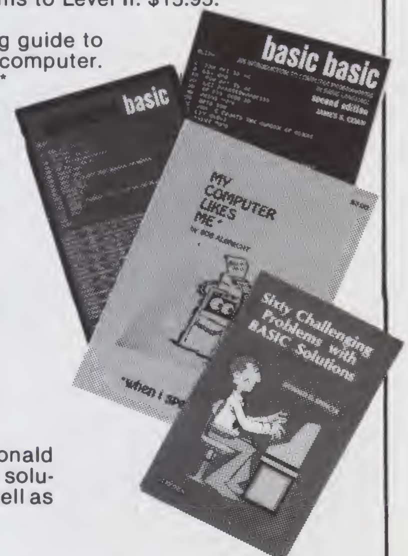
● **PROGRAMMING IN PASCAL**—BK1140—by Peter Grogono. The computer programming language PASCAL was the first language to embody in a coherent way the concepts of structured programming, which has been defined by Edsger Dijkstra and C.A.R. Hoare. As such, it is a landmark in the development of programming languages. PASCAL was developed by Niklaus Wirth in Zurich; it is derived from the language ALGOL 60 but is more powerful and easier to use. PASCAL is now widely accepted as a useful language that can be efficiently implemented, and as an excellent teaching tool. It does not assume knowledge of any other programming language; it is therefore suitable for an introductory course. $12.95.*

● **THE BASIC HANDBOOK**—BK1174—by David Lien. This book is unique. It is a virtual ENCYCLOPEDIA of BASIC. While not favoring one computer over another, it explains over 250 BASIC words, how to use them and alternate strategies. If a computer does not possess the capabilities of a needed or specified word, there are often ways to accomplish the same function by using another word or combination of words. That's where the HANDBOOK comes in. It helps you get the most from your computer, be it a "bottom-of-the-line" micro or an oversized monster. $14.95.*

● **LEARNING LEVEL II**—BK1175—by David Lien. Written especially for the TRS-80, this book concentrates on Level II BASIC, exploring every important BASIC language capability. Updates are included for those who have studied the Level I User's Manual. Sections include: how to use the Editor, dual cassette operation, printers and peripheral devices, and the conversion of Level I programs to Level II. $15.95.*



● **BASIC NEW 2ND EDITION**—BK1081—by Bob Albrecht. Self-teaching guide to the computer language you will need to know for use with your microcomputer. This is one of the easiest ways to learn computer programming. $6.95.*



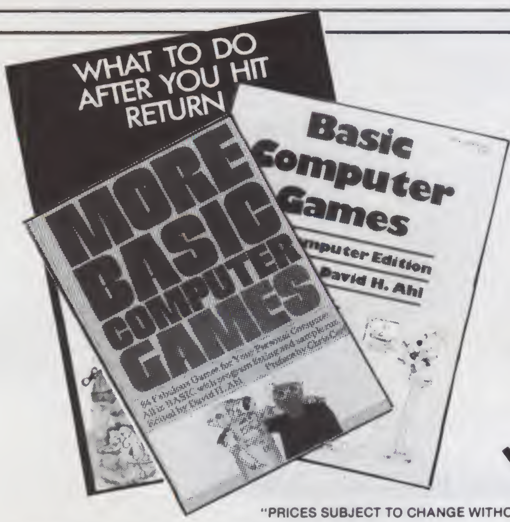
● **BASIC BASIC (2ND EDITION)**—BK1026—by James S. Coan. This is a textbook which incorporates the learning of computer programming using the BASIC language with the teaching of mathematics. Over 100 sample programs illustrate the techniques of the BASIC language and every section is followed by practical problems. This second edition covers character string handling and the use of data files. $9.45.*

● **ADVANCED BASIC**—BK1000—Applications, including strings and files, coordinate geometry, area, sequences and series, simulation, graphing and games. $9.65*.

● **MY COMPUTER LIKES ME . . . WHEN I SPEAK BASIC**—BK1039—An introduction to BASIC . . . simple enough for kids. If you want to teach BASIC to anyone quickly, this is the way to go. $3.95.*

● **SIXTY CHALLENGING PROBLEMS WITH BASIC SOLUTIONS (2nd Edition)**—BK1073—by Donald Spencer, provides the serious student of BASIC programming with interesting problems and solutions. No knowledge of math above algebra required. Includes a number of game programs, as well as programs for financial interest, conversions and numeric manipulations. $6.95.*

GAMES



● **WHAT TO DO AFTER YOU HIT RETURN**—BK1071—PCC's first book of computer games . . . 48 different computer games you can play in BASIC . . . programs, descriptions, many illustrations. Lunar Landing, Hammurabi, King, Civel 2, Qubic 5, Taxman, Star Trek, Crash, Market, etc. $10.95.*

● **BASIC COMPUTER GAMES**—BK1074—Okay, so once you get your computer and are running in BASIC, then what? Then you need some programs in BASIC, that's what. This book has 101 games for you from very simple to real buggers. You get the games, a description of the games, the listing to put in your computer and a sample run to show you how they work. Fun. Any one game will be worth more than the price of the book for the fun you and your family will have with it. $7.50.*

new!

● **MORE BASIC COMPUTER GAMES**—BK1182—edited by David H. Ahl. More fun in BASIC! 84 new games from the people who brought you *BASIC Computer Games*. Includes such favorites as Minotaur (battle the mythical beast) and Eliza (unload your troubles on the doctor at bargain rates). Complete with game description, listing and sample run. $7.50.*

"PRICES SUBJECT TO CHANGE WITHOUT NOTICE"

*Use the order card in the back of this magazine or itemize your order on a separate piece of paper and mail to Kilobaud *Microcomputing* Book Department • Peterborough NH 03458. Be sure to include check or detailed credit card information. No C.O.D. orders accepted. All orders add $1.00 handling. Please allow 4-6 weeks for delivery. Questions regarding your order? Please write to Customer Service at the above address.

FOR TOLL FREE ORDERING CALL 1-800-258-5473

WAMECO

THE COMPLETE PC BOARD HOUSE
EVERYTHING FOR THE S-100 BUSS

NEW

MEM-3, 32K STATIC RAM BOARD

* 32K of 2114's
* Full 24 line address decoding. (Or bank select like the MEM-2)
* Expandable in 1K increments. (Board disabled for unloaded RAM)
* Bi-directional capability on data lines. (Board may be used as upper or lower 8 bits on a 16 bit bi-directional buss.)
* Addressable in 8K boundary within the 64K boundary selected.
* Kits less RAM . . . . $112.95.
  With 2114L-4 . . . . $475.95.
  With 2114L-2 . . . . $549.95.
  Bare board . . . . $ 35.95.
  Available mid January.

FUTURE PRODUCTS: 80 CHARACTER VIDEO BOARD.
Z-80 CPU BOARD WITH ROM, 8 PARALLEL PORT I/O BOARD.

DEALER INQUIRIES INVITED, UNIVERSITY DISCOUNTS AVAILABLE AT YOUR LOCAL DEALER

MOST PRODUCTS FOR IMMEDIATE SHIPMENT. NO 4-8 WEEK DELAYS REQUIRED FOR OTHERS.

WMC inc.

WAMECO, INC., P. O. BOX 877 • 455 PLAZA ALHAMBRA • EL GRANADA, CA 94018 • (415) 726-6378

CALIFORNIA COMPUTER SYSTEMS

32K RAM BOARD A&T.
450 NSEC ....$579.95, 200 NSEC $629.95
16K RAM A&T.
450 NSEC ...... $255.95, 200 NSEC $285.95
64K DYNAMIC A&T.
200 NSEC $579.95
Z80 PROCESSOR A&T. $259.00
DISC CONTROLLER $339.95
APPLE IEEE INSTRUMENTATION INTERFACE KIT 7490. Kit $275.00
ARITHMETIC PROCESSOR FOR APPLE 7811A. Kit $325.95
APPLE ASYNCHRONOUS SERIAL INTERFACE 7710A. Kit $89.95
APPLE SYNCHRONOUS SERIAL INTERFACE 7712A. Kit $89.95
ALL OTHER CCS PRODUCTS AVAILABLE

SSM

PB-1 2708 & 2716 Programming Board with provisions for 4K or 8K EPROM. No external supplies required. Textool sockets. Kit.............. $143.00
CB-1A 8080 Processor Board. 2K of PROM 256 BYTE RAM power on/rest Vector Jump Parallel port with status. Kit ... $146.00 PCBD $31.95
VB-3 80x24 VIDEO BOARD. Graphics included. 4MHZ $379.95
IO-4 Two serial I/O ports with full handshaking 20/60 ma current loop: Two parallel I/O ports. Kit ....$168.00 PCBD $31.95
VB-IC 64 x 16 video board, upper lower case Greek composite and parallel video with software, S-100. Kit $143.00
CB-2 Z80 CPU BOARD. Kit $199.95
AIO APPLE SERIAL/PARALLEL $144.95
ALL OTHER SSM PRODUCTS AVAILABLE

WMC inc. WAMECO INC.

MEM-3 32K STATIC RAM 2114 24 bit addressing $32.95
FDC-1 FLOPPY CONTROLLER BOARD will drive shugart, pertek, remic 5" & 8" drives up to 8 drives, on board PROM with power boot up, will operate with CPM™ (not included). PCBD $43.95
FPB-1 Front Panel. IMSAI size, hex displays. Byte, or instruction single step. PCBD $48.50
QM-12 MOTHER BOARD, 13 slot, terminated, S-100 board only $39.95
CPU-1 8080A Processor board S-100 with 8 level vector interrupt. PCBD $28.95
RTC-1 Realtime clock board. Two independent interrupts. Software programmable. PCBD $25.95
EPM-2 2708/2716 16K/32K EPROM CARD. PCBD $28.95
QM-9 MOTHER BOARD. Short Version of QM-12. 9 Slots. PCBD $33.95
MEM-2 16K x 8 Fully Buffered 2114 Board. PCBD $28.95
PTB-1 POWER SUPPLY AND TERMINATOR BOARD. PCBD $28.95
IOB-1 SERIAL AND PARALLEL INTERFACE. 2 parallel, one serial and cassette. PCBD $28.95

2708 .... $7.50	2114L 450 NSEC $4.99
2716 ....$25.95	2114L 200 NSEC $5.99

JAN. SPECIAL SALE ON PREPAID ORDERS

(Charge cards not included on this offer)

WAMECO PCBD SALE.
MEM-2 EPM-2 . . . .$25.95
IOB-1, CPU-1 . . . .$24.95
PTB-1, RTC-1 . . . .$22.95
QMB-12 . . . . .$32.95 QMB-9 . . . . .$29.95

MIKOS PARTS ASSORTMENT WITH WAMECO AND CYBERCOM PCBDS

MEM-2 with MIKOS #7 16K ram with L2114 450 NSEC $229.95
MEM-2 with MIKOS #13 16K ram with L2114 200 NSEC $249.95
CPU-1 with MIKOS #2 8080A CPU $99.95
QM-12 with MIKOS #4 13 slot mother board $110.95
RTC-1 with MIKOS #5 real time clock........ $65.95
EMP-1 with MIKOS #10 4K 1702 less EPROMS ....$ 49.95
EPM-2 with MIKOS #11 16-32K EPROMS less EPROMS $65.95
QM-9 with MIKOS #12 9 slot mother board $99.95
FPB-1 with MIKOS #14 all parts for front panel $144.95

MIKOS PARTS ASSORTMENTS ARE ALL FACTORY MARKED PARTS. KITS INCLUDE ALL PARTS LISTED AS REQUIRED FOR THE COMPLETE KIT LESS PARTS LISTED. ALL SOCKETS INCLUDED.

LARGE SELECTION OF LS TTL AVAILABLE
PURCHASE $50.00 WORTH OF LS TTL AND GET 10% CREDIT TOWARD ADDITIONAL PURCHASES.
PREPAID ORDERS ONLY.

MIKOS

(415) 726-7593
P. O. Box 955 • El Granada, CA 94018
Please send for IC, Xistor and Computer parts list

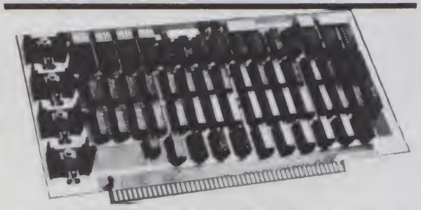
VISA or MASTERCHARGE. Send account number, interbank number, expiration date and sign your order. Approx. postage will be added. Check or money order will be sent post paid in U.S. If you are not a regular customer, please use charge, cashier's check or postal money order. Otherwise there will be a two-week delay for checks to clear. Calif. residents add 6% tax. Money back 30-day guarantee. We cannot accept returned IC's that have been soldered to. Prices subject to change without notice. **$10 minimum order. $1.50 service charge on orders less than $10.00.**

Reader Service—see page 226

WE'RE GIV

We're not selling it. The JADE Catalog is the best, and the best things in life are free. We will send you our new 1981 edition describing ove 4000 microcomputer parts, components, boards, systems, accessories

S-100 Boards



64K RAM Board - Cal Comp Sys
4 MHz, bank selectable, IEEE standard
MEM-64565A *A & T* **$449.95**

Memory Bank - Jade
4 MHz, IEEE S-100, bank selectable, 8 or 16 bit
MEM-99730K *Kit, no RAM* **$219.95**
MEM-16730K *16K kit* **$249.95**
MEM-16730A *16K A & T* **$299.95**
MEM-32731K *32K kit* **$289.95**
MEM-32731A *32K A & T* **$339.95**
MEM-48732K *48K kit* **$324.95**
MEM-48732A *48K A & T* **$374.95**
MEM-64733K *64K kit* **$359.95**
MEM-64733A *64K A & T* **$409.95**
MEM-99730B *Bare board* **$55.00**

ExpandoRAM II - SD Systems
4 MHz RAM board expandable from 16K to 256K
MEM-16630A *16K kit* **$289.95**
MEM-16630A *16K Jade A & T* **$339.95**
MEM-32631K *32K kit* **$329.95**
MEM-32631A *32K Jade A & T* **$379.95**
MEM-48632K *48K kit* **$369.95**
MEM-48631A *48K Jade A & T* **$419.95**
MEM-64633K *64K kit* **$409.95**
MEM-64633A *64K Jade A & T* **$459.95**

32K STATIC RAM - Jade
2 or 4 MHz expandable static RAM board uses 2114L's
MEM-16151K *16K 4 MHz kit* **$169.95**
MEM-16151A *16K 4 MHz A & T* **$224.95**
MEM-32151K *32K 4 MHz kit* **$299.95**
MEM-32151A *32K 4 MHz A & T* ... **$349.95**

16K STATIC RAM - Cal Comp Sys
2 or 4 MHz 16K static RAM - a real memory bargain
MEM-16160A *16K 2 MHz A & T* ... **$279.00**
MEM-16162A *16K 4 MHz A & T* ... **$309.00**
MEM-16160B *Bare board* **$29.95**

THE BIG Z* - Jade
2 or 4 MHz switchable Z-80 CPU with serial I/O*
CPU-30201K *Kit* **$145.00**
CPU-30201A *A & T* **$199.00**
CPU-30200B *Bare board* **$35.00**

SBC-100 - SD Systems
2.5 MHz Z-80 CPU with serial & parallel I/O ports*
CPC-30100K *Kit* **$299.95**
CPC-30100A *Jade A & T* **$369.95**

SBC-200 - SD Systems
4 MHz Z-80 CPU with serial & parallel I/O ports*
CPC-30200K *Kit* **$339.95**
CPC-30200A *Jade A & T* **$499.95**

CB2 - S.S.M.
2 or 4 MHz switchable Z-80 CPU with RAM, ROM, & I/O*
CPU-30300K *Kit* **$239.95**
CPC-30300A *A & T* **$299.95**

2810 Z-80* CPU - Cal Comp Sys
2/4 MHz Z-80A CPU w/serial I/O port*
CPU-30400A *A & T* **$275.00**

DOUBLE-D - Jade
Double density controller with the inside track
IOD-1200K *Kit* **$299.95**
IOD-1200A *8" A & T* **$389.95**
IOD-1205A *5¼" A & T* **$389.95**
IOD-1200B *Bare board* **$65.00**

DOUBLE DENSITY - Cal Comp Sys
5¼" or 8" disk controller with free CP/M 2.2
IOD-1300A *A & T* **$374.95**

VERSAFLOPPY II - SD Systems
New double density controller for both 8" & 5¼"
IOD-1160K *Kit* **$379.95**
IOD-1160A *Jade A & T* **$439.95**

S.P.I.C. - Jade
Our new I/O card with 2 SIO's, 4 CTC's, and 1 PIO
IOI-1045K *2 CTC's, 1 SIO, 1 PIO* .. **$199.00**
IOI-1045A *A & T* **$259.00**
IOI-1046K *4 CTC's, 2 SIO's, 1 PIO* **$259.00**
IOI-1046A *A & T* **$319.00**
IOI-1045B *Bare board w/ manual* ... **$59.95**
IOI-1045D *Manual only* **$20.00**

I/O-4 - S.S.M.
2 serial I/O ports plus 2 parallel I/O ports
IOI-1010K *Kit* **$179.95**
IOI-1010A *A & T* **$259.95**
IOI-1010B *Bare board* **$35.00**

100K DAY CLOCK - Mtn Hardware
Crystal controlled S-100 clock with NiCad backup
IOK-1400A *A & T* **$329.95**

SB1 - S.S.M.
15 Hz to 25K Hz music synthesizer for S-100
IOS-1005K *Kit* **$239.95**
IOS-1005A *A & T* **$299.95**

TB-4 - Mullen
Extremely versatile extender board with logic probe
TSX-180K *Kit* **$55.00**
TSX-180A *A & T* **$75.00**

S-100 EXTENDER - Cal Comp Sys
Puts problem boards within easy reach
TSX-160A *A & T* **$24.95**

VIDEO BOARD - Jade
64 x 16 assembled & tested S-100 video board
IOV-1050B *Bare board* **$25.00**
IOV-1050K *Kit* **$99.95**
IOV-1050A *A & T sale price* **$139.95**

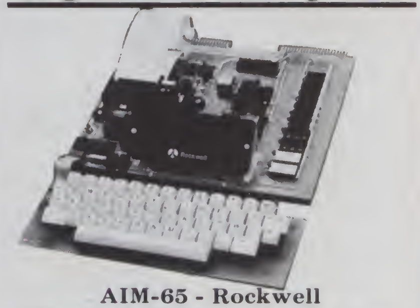
VDB-8024 - SD Systems
80 x 24 I/O mapped video board with keyboard I/O
IOV-1020K *Kit* **$399.95**
IOV-1020A *Jade A & T* **$459.95**

VB3 - S.S.M.
80 x 24 or 80 x 48 memory mapped with graphics
IOV-1095K *Kit, 4 MHz* **$399.95**
IOV-1095A *A & T, 4 MHz* **$464.95**
IOV-1096K *80 x 48 upgrade, 4 MHz* . **$89.00**

PB-1 - S.S.M.
2708, 2716 EPROM board with built-in programmer
MEM-99510K *Kit* **$159.95**
MEM-99510A *A & T* **$239.95**

PROM-100 - SD Systems
2708, 2716, 2732, 2758, & 2516 EPROM programmer
MEM-99520K *Kit* **$219.95**
MEM-99520A *Jade A & T* **$269.95**

Single Board Computers



AIM-65 - Rockwell
6502 computer with printer, display, & keyboard
CPK-50165 *1K AIM* **$374.95**
CPK-50465 *4K AIM* **$449.95**
SFK-74600008E *8K BASIC ROM* ... **$99.95**
SFK-64600004E *4K assembler ROM* **$84.95**
PSX-030A *Power supply* **$64.95**
ENX-000002 *Enclosure* **$49.95**
4K AIM, 8K BASIC, power supply, & enclosure
Special package price **$625.00**

Z-80* STARTER KIT - SD Systems
Z-80 computer with RAM, ROM, I/O, & keyboard*
CPS-30010K *Kit* **$369.95**
CPS-30010A *Jade A & T* **$459.95**

Motherboards

ISO-BUS - Jade
Silent, simple, and on sale - a better motherboard
6 Slot (5¼" x 8⅝")
MBS-061B *Bare board* **$19.95**
MBS-061K *Kit* **$39.95**
MBS-061A *A & T* **$49.95**
12 Slot (9¼" x 8⅝")
MBS-121B *Bare board* **$29.95**
MBS-121K *Kit* **$69.95**
MBS-121A *A & T* **$89.95**
18 Slot (14½" x 8⅝")
MBS-181B *Bare board* **$49.95**
MBS-181K *Kit* **$99.95**
MBS-181A *A & T* **$139.95**

Mainframes

MAINFRAME - Cal Comp Sys
12 slot S-100 mainframe with 20 amp power supply
ENC-112105 *Kit* **$309.95**
ENC-112106 *A & T* **$349.95**

DISK MAINFRAME - NNC
Dual 8" drive cutouts with 8 slot motherboard
ENS-112320 *with 30 amp p.s.* **$699.95**

Video Monitors

9" B & W MONITOR - A.P.F.
High quality, high resolution video monitor
VDM-750900 *9" monitor* **$149.9**

13" COLOR MONITOR - Zenith
The hi res color you've been promising yourself
VDC-201301 .... **$449.0**

12" GREEN SCREEN - NEC
20 MHz, P31 phosphor video monitor with audio
VDM-651200 *12" monitor* **$249.9**

SUP'R'MOD II - M & R Assoc
Color or B & W TV interface recommended for Apple
IOR-5050A *A & T* **$29.9**

NG IT AWAY

peripherals, and software. **All you have to do is ask for it.** Just circle our inquiry number on the reader service card in the rear of this magazine and we will send you the best. **It's free and it's easy.**

Accessories for Apple

16K MEMORY UPGRADE
Add 16K of RAM to your TRS-80, Apple, or Exidy

MEX-16100K *TRS-80 kit* **$39.95**
MEX-16101K *Apple kit* **$39.95**
MEX-16102K *Exidy kit* **$39.95**

DISK DRIVE for APPLE
5¼" disk drive with controller for your Apple

MSM-12310C *with controller* **$475.00**
MSM-123101 *w/out controller* **$375.00**

8" DRIVES for APPLE
Controller, DOS, two 8" drives, cabinet, & cable

Special package price **$1475.00**

AIO - S.S.M.
Parallel & serial interface for your Apple

IOI-2050K *Kit* **$159.00**
IOI-2050A *A & T* **$199.00**

PRINTER INTERFACE - Cal Comp Sys
Centronics type I/O card w/ firmware

IOI-2041A *A & T* **$99.95**

APPLE CLOCK - Cal Comp Sys
Real time clock w/battery back-up

IOK-2100A *A & T* **$119.95**

SUPERTALKER - Mtn Hardware
Speech recognition/synthesizer w/speaker & mike

IOS-2015A *A & T* **$275.00**

Z-80* CARD for APPLE
Z-80 CPU card with CP/M for your Apple*

CPX-30800A *A & T* **$279.95**

MICROMODEM - D.C. Hayes
Auto answer/dial modem card for Apple or S-100

IOM-2010A *Apple modem* **$349.95**
IOM-1100A *S-100 modem* **$375.00**

Micronet Modem - Micromate
Direct connect w/ all Micromodem II features

IOM-2020A *Best Apple modem* **$275.00**

EPROM ERASERS
Spectronics hi intensity industrial eraser

XME-3100 *Without timer* **$69.95**
XME-3101 *With timer* **$94.50**

L.S. Engineering UV eraser for up to 48 EPROMs

XME-3200 *A & T* **$39.95**

8" Disk Drive Sale

JADE's new dual disk sub-assemblies include: Handsome metal cabinet with proportionally balanced air flow system, rugged dual drive power supply, cooling fan, cable kit, lighted power switch, approved fuse assembly, line cord, Never-Mar rubber feet, and all necessary hardware to mount 2-8" disk drives - it's all American made, guaranteed for six monthes, and it's in stock!

Dual 8" Sub-Assembly Cabinet

END-000421 *Cabinet kit* **$225.00**
END-000420 *Bare cabinet* **$59.95**

Single sided, double density disk drive sub-system

END-000423 *Kit w/2 8" drives* **$995.00**
END-000424 *A & T w/2 8" drives* **$1195.00**

Double sided, double density disk drive sub-system

END-000426 *kit w/2 8" drives* **$1495.00**
END-000427 *A & T w/2 8" drives* **$1695.00**

8" DISK DRIVES
Highly reliable double density floppy disk drives

Shugart 801R single sided, double density

MSF-10801R *SA-801R* **$425.00**
Special Sale Price .... ***2 for $790.00***

Siemens FDD100-8D2 single sided, double density

MSF-201120 *6 mo warranty* **$395.00**
Special sale price .... ***2 for $750.00***

Qume Datatrak 8 double sided, double density

MSF-750080 *SA-851R compatible* .. **$625.00**
Special sale price .... ***2 for $1198.00***

JADE DISK PACKAGE
Double-D controller kit, two 8" double density drives CP/M 2.2, cabinet, power supply, & cables

Special package price **$1395.00**

DISKETTES - Jade
Bargain prices on magnificent magnetic media

5¼" single sided, single density, box of 10

MMD-5110103 *Soft sector* **$27.95**
MMD-5111003 *10 sector* **$27.95**
MMD-5111603 *16 sector* **$27.95**

5¼" double sided, double density, box of 10

MMD-5220103 *Soft sector* **$39.95**

8" single sided, single density, box of 10

MMD-8110103 *Soft sector* **$33.95**

8" single sided, double density, box of 10

MMD-8120103 *Soft sector* **$39.95**

8" double sided, double density, box of 10

MMD-8220103 *Soft sector* **$57.95**

NOVATION CAT
300 baud, auto answer/originate acoustic modem

IOM-5200A *Special sale price* **$139.00**

D-CAT 300 baud, direct connect modem

IOM-5201A *Special sale price* **$179.00**

* Z-80, Z-80A, and the letter Z are recognized trademarks of Zilog, Inc. *CP/M is a registered trademark of Digital Research Corp. *CBASIC is a trademark of Compiler Systems, Inc.

Printers

SPINWRITER - NEC
65 cps, bi-directional, letter quality with tractor

PRD-55510 *with 16K buffer* **$2595.00**

BASE 2 - Impact Printer
132 cps, bi-directional, tractor feed, & graphics

PRM-13100 .... **$649.00**

DP-9501 - Anadex
9 x 11 dot matrix, 220 column, 200 cps, & graphics

PRM-10501 *Standard DP-9501* **$1395.00**
PRM-10511 *with graphics & 2K* .. **$1450.00**

MICROPROCESSORS

Z-80	10.95
Z-80A	12.95
6502	11.50
6800	11.95
6802	17.95
6809	39.95
8035	24.00
8080A	6.59
8085	15.95
8748	59.95

Z-80 SUPPORT

3881 *PIO*	9.50
3881-4 *PIO-4 MHz*	14.50
3882 *CTC*	9.50
3882-4 *CTC-4 MHz*	14.95
3883 *SIO*	29.50
3884 *SIO*	49.50

6800 SUPPORT

6821P	5.95
6828P	11.95
6834P	22.50
6840P	18.75
6850P	4.80
6852P	5.79
6875L	7.40
68488P	25.00

BAUD RATE GENERATORS

MC14411	10.00
1.843 MHz xtal	4.95

UARTS

AY5-1013A	5.25
AY3-1014A	8.25
TR1602B	5.25
TMS6011	5.95
IM6402	9.00

PROMS

2708 *450ns*	8.25
10 for $59.00	
2716 *12,5v*	12.95
2716 *5v*	12.95
10 for $99.00	
2532 *5v*	39.95
2732 *5v*	39.95
2758 *5v*	19.95

RAMS

21L02 *2 MHz*	1.25
21L02A *4 MHz*	1.50
2114L *2 MHz*	3.75
2114LA *4 MHz*	3.95
2147 *70ns*	39.95
4116	4.95
4164 *64K x1*	175.00
5257 *2 MHz*	6.75
5257A *4 MHz*	7.25
MK4118	18.95

SUPPORT DEVICES

8212	4.95
8214	4.65
8216	2.95
8224	3.25
8224-4	10.95
8226	3.85
8228	4.95
8238	4.95
8243	8.00
8250	14.95
8251	6.50
8253	13.95
8255	6.50
8257	19.95
8259	17.95
8275	49.95
8279	15.95

PLACE ORDERS TOLL FREE

Continental U.S. **800-421-5500**
Inside California **800-262-1710**

For Technical Inquiries or Customer Service call
213-973-7707

JADE
48 **Computer Products**

4901 W. Rosecrans, Hawthorne, Ca 90250

TERMS OF SALE: Cash, checks, credit cards, or Purchase Orders from qualified firms and institutions. Minimum order $15.00. California residents add 6% tax. Minimum shipping and handling charge $2.50 **Pricing and availability subject to change without notice.**

✓ Reader Service—see page 226

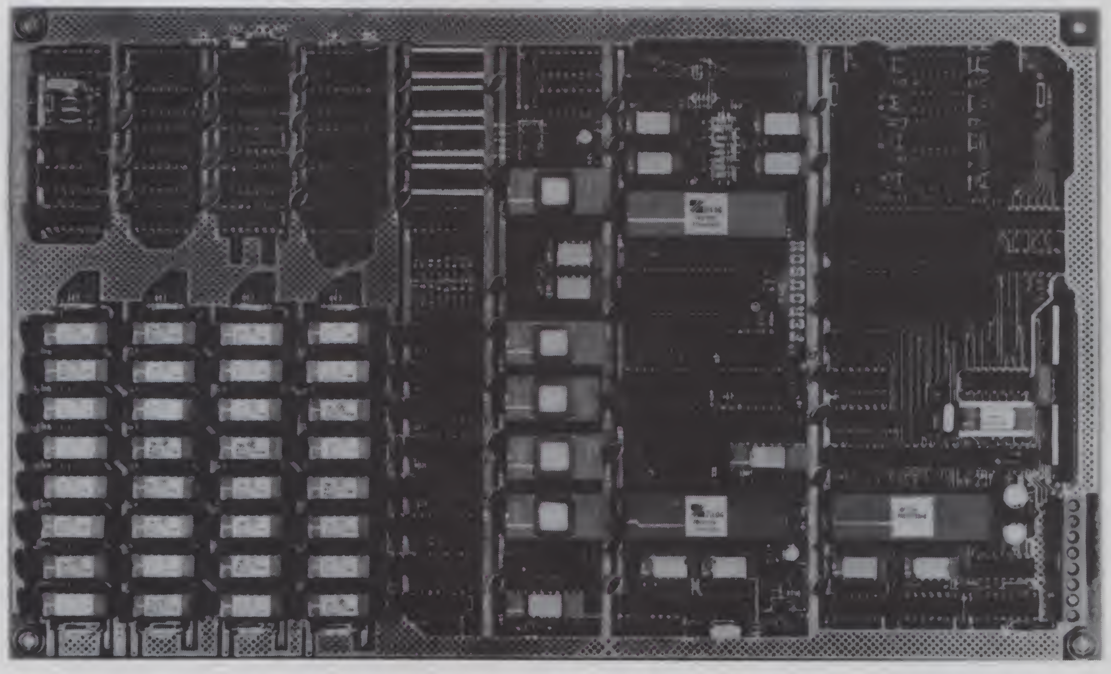
"THE BIG BOARD"

NEW! NEW!

OEM - INDUSTRIAL - BUSINESS - SCIENTIFIC

SINGLE BOARD COMPUTER KIT!

Z-80 CPU! 64K RAM!



PARTIAL KIT
For All Sockets Installed
And Soldered Add $50.

U.K. AND EUROPE: CONTACT MACLIN-ZAND LTD. 38 MOUNT PLEASANT — LONDON WCIX OAP

THE FERGUSON PROJECT: Three years in the works, and maybe too good to be true. A tribute to hard headed, no compromise, high performance, American engineering! The Big Board gives you all the most needed computing features on one board at a very reasonable cost. The Big Board was designed from scratch to run the latest version of CP/M*. Just imagine all the off-the-shelf software that can be run on the Big Board without any modifications needed! Take a Big Board, add a couple of 8 inch disc drives, power supply, an enclosure, C.R.T., and you have a total Business System for about 1/3 the cost you might expect to pay.

$649.00** (64K KIT BASIC I/O)

SIZE: 8½ x 13¾ IN.
SAME AS AN 8 IN. DRIVE.
REQUIRES: +5V @ 3 AMPS
+ - 12V @ .5 AMPS.

FULLY SOCKETED!

FEATURES: (Remember, all this on one board!)

64K RAM

Uses industry standard 4116 RAM'S. All 64K is available to the user, our VIDEO and EPROM sections do not make holes in system RAM. Also, very special care was taken in the RAM array PC layout to eliminate potential noise and glitches.

Z-80 CPU

Running at 2.5 MHZ. Handles all 4116 RAM refresh and supports Mode 2 INTERUPTS. Fully buffered and runs 8080 software.

SERIAL I/O (OPTIONAL)

Full 2 channels using the Z80 SIO and the SMC 8116 Baud Rate Generator. FULL RS232! For synchronous or asynchronous communication. In synchronous mode, the clocks can be transmitted or received by a modem. Both channels can be set up for either data-communication or data-terminals. Supports mode 2 Int. Price for all parts and connectors: $85.

BASIC I/O

Consists of a separate parallel port (Z80 PIO) for use with an ASCII encoded keyboard for input. Output would be on the 80 x 24 Video Display.

24 x 80 CHARACTER VIDEO

With a crisp, flicker-free display that looks extremely sharp even on small monitors. Hardware scroll and full cursor control. Composite video or split video and sync. Character set is supplied on a 2716 style ROM, making customized fonts easy. Sync pulses can be any desired length or polarity. Video may be inverted or true. 5 x 7 Matrix - Upper & Lower Case

FLOPPY DISC CONTROLLER

Uses WD1771 controller chip with a TTL Data Separator for enhanced reliability. IBM 3740 compatible. Supports up to four 8 inch disc drives. Directly compatible with standard Shugart drives such as the SA800 or SA801. Drives can be configured for remote AC off-on. Runs CP/M* 2.2.

TWO PORT PARALLEL I/O (OPTIONAL)

Uses Z-80 PIO. Full 16 bits, fully buffered, bi-directional. User selectable hand shake polarity. Set of all parts and connectors for parallel I/O: $29.95

REAL TIME CLOCK (OPTIONAL)

Uses Z-80 CTC. Can be configured as a Counter on Real Time Clock. Set of all parts: $14.95

SYSTEM COMPARISON

64K RAM KIT	$370.00
80 x 24 Video Kit	365.00
Floppy Disk Controller Kit	235.00
Z-80 CPU Kit	185.95
SER & PAR. I/O	129.95
S-100 Mother Board	45.00
SUB TOTAL	$1330.90

Talk about bangs per buck! The prices shown for S100 kits were taken from the July 1980 BYTE. This will give some basis for comparison between the Big Board and a similar system implementation on the S100 Buss.

CP/M* 2.2 FOR BIG BOARD

The popular CP/M* D.O.S. modified by MICRONIX SYSTEMS to run on Big Board is available for $150.00.

PC BOARD

Blank PC Board with Rom Set and Full Documentation. $195.00

PFM 3.0 2K SYSTEM MONITOR

The real power of the Big Board lies in its PFM 3.0 on board monitor. PFM commands include: Dump Memory, Boot CP/M*, Copy, Examine, Fill Memory, Test Memory, Go To, Read and Write I/O Ports, Disc Read (Drive, Track, Sector), and Search. PFM occupies one of the four 2716 EPROM locations provided.

Z-80 is a Trademark of Zilog.

Digital Research Computers
(OF TEXAS)

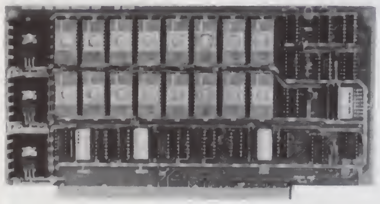
P.O. BOX 401565 • GARLAND, TEXAS 75040 • (214) 271-3538

TERMS: Initial shipments will be made approximately 3 to 5 weeks after w receive your order. VISA, MC, cash accepted. We will accept COD's (for th Big Board only) with a $75 deposit. Balance UPS COD. The $75 deposi assures your place in line for the initial production run of Big Board.

*TRADEMARK OF DIGITAL RESEARCH. NOT ASSOCIATED WITH DIGITAL RESEARCH OF CALIFORNIA, THE ORIGINATORS OF CPM SOFTWA
**1 TO 4 PIECE DOMESTIC USA PRICE.

DIGITAL RESEARCH COMPUTERS
(214) 271-3538

32K S-100 EPROM CARD
NEW!



$74.95
KIT

USES 2716's

Blank PC Board - $34

ASSEMBLED & TESTED ADD $30

SPECIAL: 2716 EPROM's (450 NS) Are $14.95 EA. With Above Kit.

KIT FEATURES:
1. Uses +5V only 2716 (2Kx8) EPROM's.
2. Allows up to 32K of software on line!
3. IEEE S-100 Compatible.
4. Addressable as two independent 16K blocks.
5. Cromemco extended or Northstar bank select.
6. On board wait state circuitry if needed.
7. Any or all EPROM locations can be disabled.
8. Double sided PC board, solder-masked, silk-screened.
9. Gold plated contact fingers.
10. Unselected EPROM's automatically powered down for low power.
11. Fully buffered and bypassed.
12. Easy and quick to assemble.

16K DYNAMIC RAM PARTIALS
LOOK! **INTEL 2108 8K X 1 RAMS** LOOK!
8 FOR $9.95 32 FOR $35
FACTORY PRIME!

Huge special purchase of INTEL Dynamic RAM's. These are 2108-4, 300NS, 8K, Ceramic DIP. The 2108 is the INTEL 2116 (16K) tested for either upper or lower 8K only. These are factory prime. Full Spec. See INTEL 1978 Cat. for details or Memory Design Handbook for application data. Both IMSAI and EXTENSYS did mfg. S-100 RAM boards using these devices. — P.S. These devices will not work in the SD EPANDORAM™. Please specify upper or lower 8K. (S1626 or S1627). A super easy RAM to interface to a Z80, 16 PIN DIP

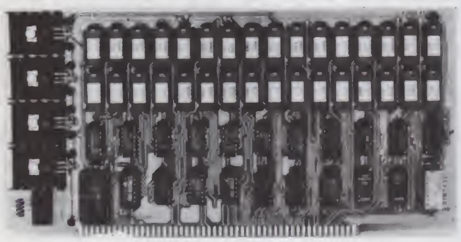
LOW POWER - 300NS 2114 RAM SALE!
FOR 4MHZ — PRICE CUT! — **8 FOR $37.50**

4K STATIC RAM'S. MAJOR BRAND, NEW PARTS.
These are the most sought after 2114's, LOW POWER and 300NS FAST.
8 FOR $37.50

16K STATIC RAM KIT-S 100 BUSS

PRICE CUT!
$199.95 KIT

FOR 4MHZ ADD $10



KIT FEATURES:
1. Addressable as four separate 4K Blocks.
2. ON BOARD BANK SELECT circuitry. (Cromemco Standard!). Allows up to 512K on line!
3. Uses 2114 (450NS) 4K Static Rams.
4. ON BOARD SELECTABLE WAIT STATES.
5. Double sided PC Board, with solder mask and silk screened layout. Gold plated contact fingers.
6. All address and data lines fully buffered.
7. Kit includes ALL parts and sockets.
8. PHANTOM is jumpered to PIN 67.
9. LOW POWER: under 1.5 amps TYPICAL from the +8 Volt Buss.
10. Blank PC Board can be populated as any multiple of 4K.

BLANK PC BOARD W/DATA-**$33**
LOW PROFILE SOCKET SET-**$12**
SUPPORT IC'S & CAPS-**$19.95**
ASSEMBLED & TESTED-**ADD $35**

OUR #1 SELLING RAM BOARD!

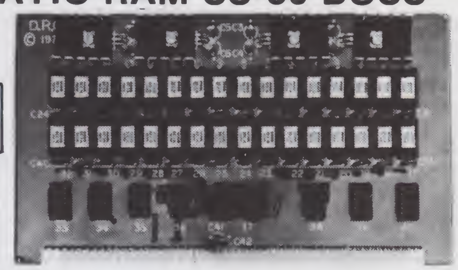
16K STATIC RAM SS-50 BUSS

PRICE CUT!

$210 KIT

FULLY STATIC!

FOR 2MHZ ADD $10



FOR SWTPC 6800 BUSS!

ASSEMBLED AND TESTED - $35

KIT FEATURES:
1. Addressable on 16K Boundaries
2. Uses 2114 Static Ram
3. Fully Bypassed
4. Double sided PC Board. Solder mask and silk screened layout.
5. All Parts and Sockets included
6. Low Power: Under 1.5 Amps Typical

BLANK PC BOARD—**$30** COMPLETE SOCKET SET—**$12**
SUPPORT IC'S AND CAPS—**$19.95**

STEREO! S-100 SOUND COMPUTER BOARD
NEW! NEW!

At last, an S-100 Board that unleashes the full power of two unbelievable General Instruments AY3-8910 NMOS computer sound IC's. Allows you under total computer control to generate an infinite number of special sound effects for games or any other program. Sounds can be called in BASIC, ASSEMBLY LANGUAGE, etc.

COMPLETE KIT! $84.95 (WITH DATA MANUAL)

BLANK PC BOARD W/DATA $31

KIT FEATURES:
* TWO GI SOUND COMPUTER IC'S.
* FOUR PARALLEL I/O PORTS ON BOARD.
* USES ON BOARD AUDIO AMPS OR YOUR STEREO.
* ON BOARD PROTO TYPING AREA.
* ALL SOCKETS, PARTS AND HARDWARE ARE INCLUDED.
* PC BOARD IS SOLDERMASKED, SILK SCREENED, WITH GOLD CONTACTS.
* EASY, QUICK, AND FUN TO BUILD. WITH FULL INSTRUCTIONS.
* USES PROGRAMMED I/O FOR MAXIMUM SYSTEM FLEXIBILITY.

Both Basic and Assembly Language Programming examples are included.

SOFTWARE:
SCL™ is now available! Our Sound Command Language makes writing Sound Effects programs a SNAP! SCL™ also includes routines for Register-Examine-Modify, Memory-Examine-Modify, and Play-Memory. SCL™ is available on CP/M* compatible diskette or 2708 or 2716. Diskette - **$24.95** 2708 - **$19.95** 2716 - **$29.95.** Diskette includes the source. EPROM'S are ORG at E000H.

4K DYNAMIC RAM BLOWOUT!
SAME AS INTEL 2107B!
4K RAMS AT AN UNBELIEVABLE 50¢ EACH!!!

Prime, new, National Semi., 1979 date coded, full spec. parts. N.S. #MM5280-5N. Same as INTEL 2107B-4, T.I. TMS4060, NEC uPD411, etc. We bought a HUGE QTY. from a West Coast Distributor at truly DISTRESS PRICES! One of the most popular and reliable RAM's ever made. These parts have been used by almost all Major Computer Main Frame Mfg. the world over! Arranged as 4K x 1, *270 NS Access Time*, 22 Pin Dip. These units DO NOT use multiplexed addressing, thus making REFRESH and other timing very simple. See INTEL MEMORY DESIGN HANDBOOK for full application notes. The NAT. SEMI. MEMORY DATA BOOK is available at most Radio Shack Stores. Prime units in original factory tubes!

(With Pin Out Data) **#5280-5N 4096 BITS x 1 270 NS ACCESS**
8 FOR $4.95 32 FOR $16
FACTORY CASE (450 PCS) — $180
Sockets Special: 22 Pin Low Profile (With Purchase of 5280's) 8 FOR $1.

COMPUTER PARTS SPECIALS

74LS175 - **.99**	8035 Intel Single Chip CPU - **5.95**
74LS240 - **1.79**	Signetics 2901 4 Bit Slice - **6.95**
74LS241 - **1.79**	AMD 2903 4 Bit Super Slice - **12.50**
74LS244 - **1.79**	AMD 29705 Dual Port RAM - **8.95**
74LS373 - **1.99**	

NEW! G.I. COMPUTER SOUND CHIP

AY3-8910. As featured in July, 1979 BYTE! A fantastically powerful Sound & Music Generator. Perfect for use with any 8 Bit Microprocessor. Contains: 3 Tone Channels, Noise Generator, 3 Channels of Amplitude Control. 16 bit Envelope Period Control, 2-8 Bit Parallel I/O. 3 D to A Converters, plus much more! All in one 40 Pin DIP. Super easy interface to the S-100 or other busses. **$11.95** **PRICE CUT!**
SPECIAL OFFER: ~~$14.95~~ each Add $3 for 60 page Data Manual.

Digital Research Computers
(OF TEXAS)
P.O. BOX 401565 • GARLAND, TEXAS 75040 • (214) 271-3538

TERMS: Add $1.50 postage. We pay balance. Orders under $15 add 75¢ handling. No C.O.D. We accept Visa and MasterCharge. Tex. Res. add 5% Tax. Foreign orders (except Canada) add 20% P & H. Orders over $50, add 85¢ for insurance.

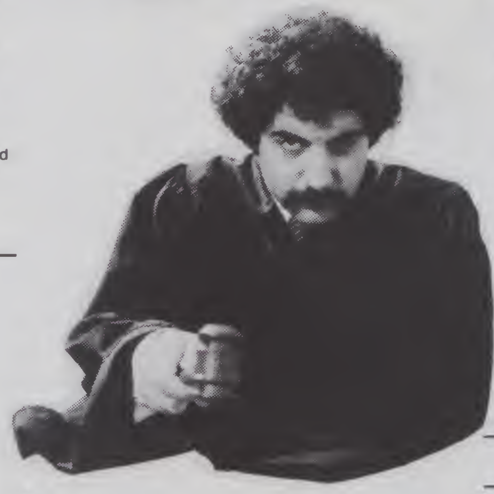
*TRADEMARK OF DIGITAL RESEARCH. NOT ASSOCIATED WITH DIGITAL RESEARCH OF CALIFORNIA, THE SUPPLIERS OF CPM SOFTWARE.

✓ Reader Service—see page 226

Printers

SEND FOR OUR FREE CATALOG

CompuMart let's you put'em on trial



with 10 day free return

ANNADEX.

132 col. printer. High Quality at a super price. Dot matrix, serial parallel & current loop standard. 180 c.p.s. Bi-Directional

Memory Integrated Circuits - Call for Qty. discounts when ordering over 50 units.

Motorola 4116 (200 Nanosecond, Plastic) **$4.50 EA.**
Fairchild 2114 (Standard Power, Plastic) **$4.50 EA.**

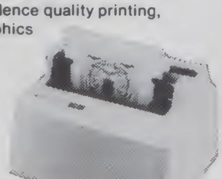
NEW From Integral Data The IDS 445 Printer

So new all we can tell you is that it's fantastic. Priced lower than the 440 and equipped with a better print head. Advanced technology strikes!

IDS 445 w/Graphics Capabilities **$894**
IDS 445 w/o Graphics Capabilities **$795**

The IDS 460

Since its introduction 5 months ago this printer has already proved itself. Features include: Correspon dence quality printing, High-resolution graphics capability, programmable print justification —a great printer.



$1,295

CENTRONICS PRINTERS

New! The incredible Model 737- Correspondence and Draft Quality Printing for Under $1,000. This is the first printer in its class to offer print quality suitable for text processing, plus the performance and application flexibility required for data processing.

737-1 (Parallel Interface)— **$899**
737-3 (Serial Interface) **$1,045**

Tractor Feed Printer- Centronics' Most Popular Model. Perfect for the needs of a small business sytesm. Recommended by Apple and Radio Shack. **$1,079**

FREE Cable with purchase of any Centronics printer.

The Omni 810 Printer from Texas Instruments

TI Omni 820 Receive-Only (RO) Package. Includes machine-mounted paper tray and cable. A compressed print option and device forms control are standard features. **$2,155**

TI Omni 820 Keyboard Send Receive (KSR) Package comes with full ASCII keyboard with numeric key-pad and an EIA cable with autospeed select. **$2,395**

NEC The First Name in Letter Quality Printers.

CompuMart offers beautiful print quality with NEC Spinwriter terminals. The Spinwriters, both KSR and RO versions, give unsurpassed hard copy output. CompuMart offers a complete range of NEC Spin-writers — Call our expert salesforce.

Send for our DEC LSI-11 Systems Catalog

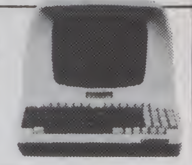
CompuMart has been selling computers by mail since 1971. Our thousands of satisfied customers rely on CompuMart for services not generally available from the others. Namely:

- Product Selection/Each product advertised by CompuMart has been evaluated by our in-house staff for best price, performance, and supplier reliability.
- Return Privilege/After receipt of our products, you are protected by CompuMart's exclusive, 10-day return priviledge good for all products except software.
- Support/Our Customer Service Dept. and expert technicians are always there to assist you by phone or at CompuMart's outlets. Our knowledgeable phone sales force can provide you with detailed information and complete product specifications.
- Phone Ordering/For added convenience, CompuMart maintains a toll-free ordering number. 1-800-343-5504.
- Phones open M T W Th F 8:30 a.m. - 7:00 p.m. Sat. 11:00 a.m. - 4:00 p.m.

Compumart will throw in a **FREE** fabric ribbon and Currier 72 element when you buy a NEC from us.

Terminals

NEW from Lear Siegler!



IT's here—IT is the difference between too smart and too dumb. IT is the new Intermediate Terminal from Lear Siegler.

We've got the following Lear Siegler Terminals In Stock at prices too low to print—**Call** for quotes.

ADM-3A Industries favorite dumb terminal for some very smart reasons.
ADM-3A.+ New from Lear Siegler. CALL!
ADM-31. The terminal that's too smart to be con sidered dumb.
ADM-42. Available with keyboard or without, semi intelligent terminal offering tremendous user flexi bility. The optional configurations are amazing. **Call for details.**

Limited Time Offer
We've slashed our Hazeltine prices again!

Hazeltine 1410	List $850	CompuMart **$749**
Hazeltine 1420	List $995	CompuMart **$825**
Hazeltine 1500	List $1095	CompuMart **$965**
Hazeltine 1510	List $1395	CompuMart **$1135**
Hazeltine 1520	List $1585	CompuMart **$1199**
Hazeltine 1552	List $1395	CompuMart **$1235**

Calculators

HEWLETT-PACKARD'S HP-41C

HP-41C Calculator **$288.00**
The System
Memory Modules. For storing programs or up to 2,000 lines of program memory **$45.00**
"Extra Smart" Card Reader. Records programs and data back onto blank mag-cards **$199.00**
The Printer. Upper and Lower case, High resolution plotting, Portable Thermal operation **$355.00**
Application Modules .... **$45.00 EACH**

Texas Instruments TI-99/4 Home Computer
Save $300 on this 16-Bit computer with monitor
TI-99/4 w/o Monitor **$659**

NOVATION CAT™
ACCOUSTIC MODEM

- Answer Originate
- 300 Baud
- Bell 108
- Low Profile Design

Looks good, works great! **$179.00**

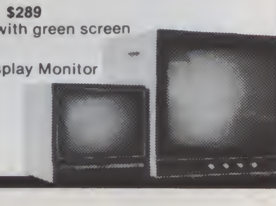
NEW! D-CAT
Direct Coupler from NOVATION **$199.**

Monitors

EXCLUSIVE from CompuMart!
Special Offer. Zenith Color Video Monitor for $379!

NEW FROM SANYO — Four Great Monitors at Low CompuMart Prices.
Sanyo's new line of CRT data display monitors are specifically designed for the display of alpha-numeric or graphic data.

9" Sanyo Monitor **$169**
12" Sanyo Monitor **$289**
12" Sanyo Monitor with green screen **$299**
13" Sanyo Color Display Monitor **$495**



Computers

We carry the most complete inventory of Apple computers, peripherals, and software. CALL!

Our Best Selling Apple System:

Save over $250 on our most popular Apple System. System includes a 48K Apple II, Apple Disk & Controller, and a Sup R Mod RF Modulator.

List: $2,020

Compumart Sale Price: $1,769

New from Apple for the Apple II:

DOS 3-3 Convert disks to 16 sector format for 23% more storage and faster access **$60**

Apple Plot. The perfect graphic complement for Visicalc.	$70
Dow Jones News & Quotes	$95
Adventure (Uses 48K)	$35
DOS Tool Kit	$75
Apple Fortran	$200
Silentype Printer w/Xface	$595
Visicalc	$149
Tax Planner	$120

From Symtech & Info Unlimited

Super Sound Generator (mono) **$159** (stereo)	$259
Light Pen	$249
X-10 Controller (plugs into paddle port)	$49

From **Personal Software**

Visicalc	$149
Desk top plan	$99

New from **Videx!** — Video Term
80 Col. x 24 line
7 x 9 matrix, plug in compatible board for the Apple II. Price **$325** without graphics EPROM. With graphics EPROM **$350.**

New from **MUSE**

The Voice	$39.95
Super Text	$99.00
Address Book	$49.95

Mountain Hardware — Expansion accessories for your Apple

Introl/X-10 System	$289
Super Talker	$299
The Music System	$545
ROM plus board w/keyboard filter	$199
Clock Calendar	$280
16 Channel A to D Converter	$350
Apple Expansion Chassis	$650
ROM Writer	$175

Miscellaneous Apple II Accessories:

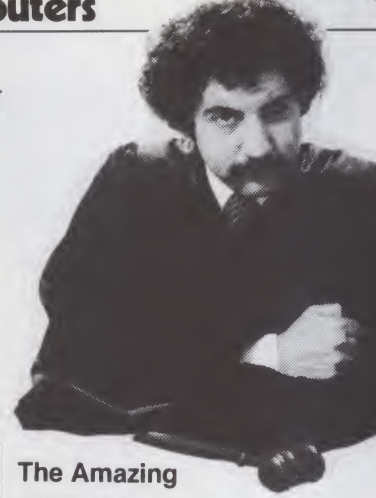
Easy Writer (80 col. need Videx)	$249
Easy Mover	$ 49
Easy Mailer	$ 69
Dysan Diskettes	ea. $ 5
S.S.M. Serial & Parallel Apple Interface	$225
ABT's Numeric Key Pad	$110

COMMODORE

Buy direct from the biggest — Compumart has delivered more Commodore computers in the U.S. than any other dealer. We were Commodore's first dealer and carry everything Commodore manufactures. In stock for immediate delivery! Call us now for low prices and special deals.

NEW FOR PET:

Visicalc (Need 32K and a disk drive) $199
Word Pro 1, **$29.95** • Word Pro 2, **$99.95** •
Word Pro 3, **$199.95** • Word Pro 4, **$299.95** •



The Amazing
HP 85

Hewlett-Packard's Personal Computer for Industry. This extremely portable computer features extended BASIC to solve your problems quickly and efficiently along with an advanced graphics system to enhance communication.

We carry H.P. Peripherals (Disk Drives to Graphics Plotters) Enhancements: (BASIC Training, General Statistics, Financial Decision, Math, Linear Programming **$95 ea.**); HP-85 Accessories: (Enhancement ROMs, ROM drawer, Overhead Transparency Kit); Supplies: (Plotter Pens, Tape Cartridges); Interface Modules (HP-IB Interface, HP-IB Interconnect Cables, Serial (RS-232C) Interface Module).

We can get your every HP peripheral made for the HP-85. **CALL FOR COMPLETE DETAILS & SPECS.**

EXIDY SUPER SALE!

We want to clean out our inventory of Exidy computers and peripherals. To do this we've priced our Exidy equipment so low you'll have to call us for prices.

ROCKWELL AIM 65

The single board development system that's perfect in the classroom or lab.

Our AIM System includes: 4K AIM with BASIC interpretor assembler, Power Supply, Cassette recorder & Enclosure **$799.**

4K AIM — 65	$499
PL65 High Level Language	$125
Paper for the AIM (roll)	$ 2.50
Rockwell's 4-slot Motherboard (SALE)	$175

CompuMart's Microflex 65 System for your AIM Includes: Adapter Buffer Module w/ 4-slot module stack, 8K RAM module, 16K PROM/ROM module, Asynchronous communications Interface, & Power Supply **$1,299**

Call or write for our complete Microflex 65 brochure

Piggy-back for the AIM.

★ CompuMart STOCKS THE COMPLETE LINE OF MATROX PRODUCTS. CALL FOR SPECS.

COMPUMART'S EXCLUSIVE ATARI SPECIALS.

3 Ways to save when you buy the Atari 800 from us. Choose from one of the following offers.

ATARI

1) **Free** 8K of memory with purchase. (So your Atari will come to you with 24K.)
2) **Free** 410 program recorder with purchase ($89.95 value).
3) $100 off Atari Disk Drive purchase.

ATARI 800 Personal Computer System —

Comes with 800 Operators Manual, 16K RAM Memory module, 10 K ROM Operating System, power supply, TV Switch Box. **$950.**

PERIPHERALS

Atari 410 Program Recorder (FREE w/purchase of Atari 800)	$ 89.95
Atari 810 Disk Drive ($100 off with purchase)	699.95
New Dual Disk double density	1499.95
825 Printer (Centronics 737)	995.00
RS232 Interface w/Cable	249.95
NEW! Light Pens	74.95
New! Visicalc for Atari	**$199.**

NEW! The PMC-80 The new 12K computer that's SOFTWARE COMPATIBLE with the TRS-80.*

Level II 16K at $645

Just think, now you can choose from the thousands of software packages already developed for the TRS-80 and run them on a computer costing only **$645.** But the PMC-80 is more than just a computer that accepts TRS-80 software. Its features include: Reads all Level II BASIC tapes. Reads all SYSTEM tapes. Full range of peripherals. Video output for monitor and TV. Optional FASTLOAD at 8000 baud. Optional Upper/Lower case. Call for complete specs.

*TRS-80 is a registered trademark of Tandy Radio Shack

IMPORTANT ORDERING INFORMATION All orders must include 4% shipping and handling. Mass. residents add 5% sales tax, Michigan residents 4% for sales tax.

TO ORDER CALL: 800-343-5504
In Mass. call 1-617-491-2700

Phones open from 8:30 a.m. to 7:00 p.m., Mon.-Fri.; 11:00 a.m. - 4:00 p.m. Sat. P.O.'s accepted from Dun & Bradstreet rated companies - shipment contingent upon receipt of signed purchase order. Sale prices valid for month of magazine date only - all prices subject to change without notice. Our Ann Arbor retail store is open 11:00 a.m. to 7:00 p.m. Tues.-Fri., 10:00 a.m. to 5:00 p.m. Saturdays. Stop by and visit.

COMPUMART

↙43

270 THIRD ST., DEPT. 133, P.O. BOX 568, CAMBRIDGE, MA. 02139

Member Computer Dealers Assoc.

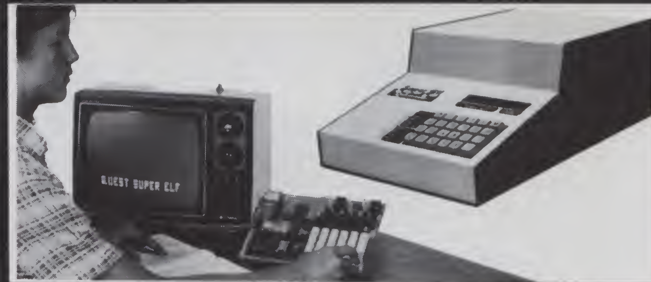
We've had a reputation for dependability since 1971

Quest ELECTRONICS

✓44

P.O. Box 4430S
Santa Clara, CA 95054
Will calls: 2322 Walsh Ave.
(408) 988-1640

Same day shipment. First line parts only. Factory tested. Guaranteed money back. Quality IC's and other components at factory prices.



INTEGRATED CIRCUITS

7400TTL	
7400N	19
7402N	20
7404N	25
7409N	23
7410N	20
7414N	70
7420N	20
7422N	39
7430N	20
7442N	58
7445N	74
7447N	58
7448N	77
7450N	20
7474N	35
7475N	49
7485N	88
7489N	1 85
7490N	43
7492N	43
7493N	43
7495N	69
74100N	1 20
74107N	35
74121N	34
74123N	59
74125N	45
74145N	77
74150N	1 20
74151N	69
74154N	2 00
74157N	69
74161N	87
74162N	1 25
74163N	87
74174N	96
74175N	95
74190N	1 15
74192N	87
74193N	85
74221N	2 75
74298N	1 65
74365N	89
74366N	89
74367N	89

74LS00 TTL	
74LS00N	35
74LS02N	35
74LS04N	55
74LS05N	40
74LS08N	45
74LS10N	45
74LS13N	55
74LS14N	1 25
74LS20N	45
74LS22N	45
74LS28N	45
74LS30N	45
74LS33N	75
74LS38N	75
74LS74N	1 25
74LS75N	1 00
74LS90N	1 00
74LS93N	1 00
74LS95N	1 10
74LS107N	65
74LS112N	65
74LS113N	45
74LS132N	89
74LS136N	65
74LS151N	1 10
74LS155N	1 10
74LS157N	1 10
74LS162N	1 65
74LS163N	1 65
74LS174N	2 00
74LS190N	1 25
74LS221N	2 50
74LS258N	1 60
74LS367N	1 65

LINEAR	
CA3045	90
CA3046	1 10
CA3081	1 80
CA3082	1 90
CA3089	2 95
LM301AN AH	35
LM305H	87
LM307N	35
LM308N	1 00
LM309K	1 50
LM311H N	90
LM317T/K	3 75
LM318	1 35
LM320K-5	1 50
LM323K-5	5 95
LM320K-12	1 50
LM320K-15	1 50
LM320T-5	1 35
LM320T-8	1 35
LM320T-12	1 35
LM320T-15	1 35
LM324N	1 40
LM339N	1 00
LM340K-5	1 35
LM340K-8	1 35
LM340K-12	1 35
LM340K-15	1 35
LM340K-24	1 35
LM340T-5	1 25
LM340T-8	1 25
LM340T-12	1 25
LM340T-15	1 25
LM340T-18	1 25
LM340T-24	1 25
LM350	7 50
LM377	3 50
LM379	5 00
LM380N	1 00
LM381	1 60
LM382	1 60
LM703H	65
LM709H	28
LM723H/N	50
LM733N	85
LM741CH	35
LM741N	38
LM747H/N	75
LM748N	35
LM1303N	1 75
LM1304	1 10
LM1305	1 27
LM1307	2 00
LM1310	2 75
LM1458	47
LM1812	7 50
LM1889	3 00
LM2111	1 75
LM2902	2 25
LM3900N	60
LM3905	1 75
LM3909N	95
MC1458V	50
NE550N	1 00
NE555V	39
NE556A	85
NE565A	1 00
NE566V	1 50
NE567V	1 00
NE570B	4 75
78L05	60
78L08	60
78M05	85
75108	1 75
75491CN	50
75492CN	55
75494CN	89

A to D CONVERTER	
80388	4 50
8700CJ	13 95
8701CN	22 00
8750CJ	13 95
LD130	9 95
9400CJV/F	7 40
ICL7103	9 50
ICL7107	14 25

CMOS	
CD4000	25
CD4001	35
CD4002	35
CD4006	1 10
CD4007	35
CD4008	28
CD4009	45
CD4010	45
CD4011	35
CD4012	28
CD4013	47
CD4014	1 25
CD4015	1 00
CD4016	55
CD4017	1 05
CD4018	94
CD4019	45
CD4020	1 02
CD4021	1 35
CD4022	1 10
CD4023	28
CD4024	75
CD4025	28
CD4026	2 50
CD4027	66
CD4028	85
CD4029	1 35
CD4030	45
CD4035	1 35
CD4040	1 35
CD4042	85
CD4043	85
CD4044	85
CD4046	1 67
CD4049	45
CD4050	49
CD4051	1 13
CD4060	1 42
CD4066	71
CD4068	40
CD4069	40
CD4070	50
CD4071	45
CD4072	45
CD4073	45
CD4075	45
CD4076	1 65
CD4078	40
CD4081	35
CD4082	35
CD4116	47
CD4490	5 50
CD4507	1 00
CD4508	4 25
CD4510	1 02
CD4511	94
CD4515	2 52
CD4516	1 10
CD4518	1 02
CD4520	1 02
CD4527	1 51
CD4528	79
CD4553	3 50
CD4566	2 25
CD4583	2 35
CD4585	1 10
CD40192	3 00
74C00	35
74C04	40
74C10	35
74C14	1 95
74C20	35
74C30	35
74C48	2 25
74C74	85
74C76	1 75
74C90	1 75
74C93	1 75
74C154	3 00
74C160	2 00
74C175	1 35
74C192	2 25
74C221	2 50
74C905	6 00
74C906	75
74C914	1 95
74C922	6 00
74C923	6 00
74C925	7 50
74C926	6 95
74C927	6 95

INTERFACE	
8095	65
8096	65
8097	65
8098	65
8T09	1 25
8T10	4 50
8T13	3 00
8T20	5 50
8T23	3 10
8T24	3 50
8T25	3 20
8T26	1 69
8T28	2 75
8T97	1 69
8T98	1 69

MOS/MEMORY RAM	
2101-1	2 95
2102-1	95
2102AL-4	1 34
2102AN-2L	1 60
2104A-4	4 95
2107B-4	3 75
2111-1	3 75
2112-2	3 95
2114L 300ns	6 75
2114L 450ns	5 95
4116 200ns	7 95
8/4116 200ns	49 00
25138	6 30
MM5262	40
MM5280	3 00
MM5320	9 95
MM5330	5 94
PD411D-3	4 00
PD411D-4	5.00
P5101L	8 95
4200A	9 95
82S25	2 90
91L02A	1 50
HD0165-5	8 95
MM57100	4 50
GIAY38500-1	9 95
MCM66751A	9 95
9368	3 50
410D	10 00
416	16 00

CLOCKS	
MM5311	5 50
MM5312	3 90
MM5314	3 90
MM5369	2 10
MM5841	14 45
MM5865	7 95
CT7010	8 95
CT7015	8 95
MM5375AA/N	3 90
MM5375AG/N	4 90
7205	16 50
7207	7 50
7208	15 95
7209	4 95
DS0026CN	3 75
DS0056CN	3 75
MM53104	2 50

MICROPROCESSOR	
6502	10 95
6504	9 95
6522	9 95
6800	6 95
6802	11 95
6820	4 95
6850	5 95
8080A	5 95
8085	12 95
8086	75 00
Z80	9 95
Z80A	11 95
8212	2 90
8214	3 95
8216	2 90
8224	3 45
8228	4 95
8251	6 95
8253	15 00
8255	5 75
8257	10 95
8259	14 95
1802CP plas	13 95
1802DP plas	17 95
1861P	11 50
CDP1802CD	28 95
CDP1802D	35 00
CDP1861	15 95

UART/FIFO	
AY5-1013	5 50
AY5-1014	7 50
3341	6 95

PROM	
1702A	4 95
2513B upper case	8 75
2708	7 75
2716T1	18 00
2716 Intel	23 00
8/2716 Intel	160 00
2732	65 00
2758	22 50
8741A	60 00
8748	65 00
8748-8	60 00
8755A	55 00
N82S23	2 95
N82S123	6 50
N82S126	3 75
N82S129	8 50
N82S131	8 50
N82S136	8 75
N82S137	8 75
DM8577	2 90
8223	2 90

CONNECTORS	
30 pin edge	2 50
44 pin edge	2 75
100 pin edge	4 50
100 pin edge WW	5 25

IC SOCKETS

Solder Tin Low Profile			
PIN	1 UP	PIN	1UP
8	15	22	30
14	14	24	35
16	16	28	42
18	27	36	58
20	29	40	57

2 level 14 pin ww 20

WIRE WRAP LEVEL 3

PIN		PIN	
14	32	24	86
16	33	28	1 00
18	57	40	1 23

CRYSTALS	
1 MHz	4 50
2 MHz	4 50
4 MHz	4 25
5 MHz	4 25
10 MHz	4 25
18 MHz	3 90
20 MHz	3 90
32 MHz	3 90
32768 Hz	4 00
1 8432 MHz	4 50
3 5795 MHz	1 20
2 0100 MHz	1 95
2 097152 MHz	4 50
2 4576 MHz	4 50
3 2768 MHz	4 50
5 0688 MHz	4 50
5 185 MHz	4 50
5.7143 MHz	4 50
6 5536 MHz	4 50
14 31818 MHz	4 25
18 432 MHz	4 50
22 1184 MHz	4 50

KEYBOARD ENCODERS	
AY5-2376	$12 50
AY5-3600	17 95
AY5-9100	10 50
AY5-9200	16 50
74C922	5 50
74C923	5 50
HD0165-5	6 95
AY5-9400	10 50

D Connectors RS232	
DB25P	3 62
DB25S	5 20
Cover	1 67
DE9S	1 95
DA15P	2 10
DA15S	3 10
Complete Set	9 50

Hickok 3½ Digit LED multimeter	**74.95**
Stopwatch Kit	**26.95**
Auto Clock Kit	**17.95**
Digital Clock Kit	**14.95**

8K/16K Eprom Kit	
(less PROMS)	$89 00
Motherboard	$39 00
Extender Board	$15 00

RESISTORS ¼ watt 5%
10 per type 03
25 per type 025
100 per type 015
1000 per type 012
350 piece pack
5 per type 6 75
½ watt 5% per type 05

Televideo Terminal	
Model 912	$845 00
Model 920	$945 00

Tiny Basic Experimenters Kit **$10.00**

BSR Controller $39.95
Connect your computer to the BSR Home Control System. Computer controlled ultrasonic transmitter for your BSR. Software for 1802 user

KEYBOARDS	
56 key ASCII keyboard kit	$67 50
Fully assembled	77 50
53 key ASCII keyboard kit	60 00
Fully assembled	70 00
Enclosure Plastic	14 95
Metal Enclosure	29 95

LEDS	
Red T018	15
Green, Yellow T018	20
Jumbo Red	20
Green, Orange, Yellow Jumbo	25
Cliplite LED Mounting Clips	8/$1.25

(specify red, amber, green, yellow, clear)

CONTINENTAL SPECIALTIES in stock
Complete line of breadboard test equip
MAX-100 8 digit Freq. Ctr. $128.95

OK WIRE WRAP TOOLS in stock
Portable Multimeter $18.00
Complete line of AP Products in stock

SPECIAL PRODUCTS	
MM5865 Stopwatch Timer with 10 pg spec	9 00
PC board	7 50
Switches Mom Pushbutton	27
3 pos. slide	25
Encoder HD0165-5	6 95
Paratronics 100A Logic Analyzer	**$425.00**
Model 10 Trigger Expander Kit	$229 00
Model 150 Bus Grabber Kit	$369 00
Clock Calendar Kit	**$23.95**
2.5 MHz Frequency Counter Kit	**$37.50**
30 MHz Frequency Counter Kit	**$47.75**

TRANSFORMERS	
6V 300 ma	3 25
12 Volt 300 ma transformer	1 25
12 6V CT 600 ma	3 75
12V 250 ma wall plug	2 95
12V CT 250 ma wall plug	3 50
24V CT 400 ma	3 95
10V 1 2 amp wall plug	4 85
12V 8 amp	12 95
12V 500 ma wall plug	4 75
12V 1 amp wall plug	6 50
10/15 VAC 8/16 VA wall plug	9 75

DISPLAY LEDS			
MAN1	CA	270	2 90
MAN3	CC	125	39
MAN72/74	CA/CA	300	1 00
DL704	CC	300	1.25
DL707/DL707R	CA	300	1 00
DL727/728	CA/CC	500	1 90
DL747/750	CA/CC	600	1 95
DL750	CC	600	1 95
FND359	CC	357	70
FND500/507	CC/CA	500	1 35
FND503/510	CC/CA	500	90
FND800/807	CC/CA	800	2 20
3 digit Bubble			60
4 digit Bubble			80
DG8 Fluorescent			1.75
DG10 Fluorescent			1 75
10 digit display			1 25
7520 Clairex photocells			39
TIL311 Hex			9 50
MAN3640	CC	30	1 10
MAN4610	CA	40	1.20
MAN4640	CC	40	1 20
MAN4710	CA	40	95
MAN4740	CC	40	1.20
MAN6640	CC	56	2 95
MAN6710	CA	60	1 35
MAN6740	CC	60	1 35

MA1002A	**8.95**
MA1002E	**8.95**
MA1012A	**8.95**
102P3 transformer	**2.25**
MA1012A Transformer	**2.25**

RCA Cosmac 1802 Super Elf Computer $106.95

Compare features before you decide to buy any other computer. There is no other computer on the market today that has all the desirable benefits of the **Super Elf** for so little money. The Super Elf is a small single board computer that does many **big** things. It is an excellent computer for training and for learning programming with its machine language and yet it is easily **expanded** with **additional memory, Full Basic, ASCII Keyboards, video character generation, etc.**

Before you buy another small computer, see if it includes the following features: ROM monitor; State and Mode displays; Single step; Optional address displays; Power Supply; Audio Amplifier and Speaker; Fully socketed for all IC's; Real cost of in warranty repairs; Full documentation.

The **Super Elf** includes a **ROM** monitor for program loading, editing and execution with **SINGLE STEP for program debugging** which is not included in others at the same price. With **SINGLE STEP** you can see the microprocessor chip operating with the **unique Quest** address and data bus displays **before, during** and **after** executing instructions. Also, CPU mode and instruction cycle are decoded and displayed on 8 LED indicators.

An RCA 1861 **video graphics** chip allows you to connect to your own TV with an inexpensive video modulator to do graphics and games. There is a **speaker system** included for writing your own music or using many music programs already written. The speaker amplifier may also be used to drive relays for control purposes.

A 24 key **HEX keyboard** includes 16 HEX keys plus **load, reset, run, wait, input, memory protect, monitor select** and **single step**. Large, on board displays provide output and optional **high and low address**. There is a 44 pin standard connector slot for PC cards and a 50 pin connector slot for the Quest Super Expansion Board. Power supply and sockets for **all** IC's are included in the price plus a detailed 127 pg. instruction manual which now includes over 40 pgs. of software info. including a series of lessons to help get you started and a music program and graphics target game. Many schools and universities are using the Super Elf as a course of study. OEM's use it for training and R&D.

Remember, other computers only offer Super Elf features at additional cost or not at all. **Compare before you buy. Super Elf Kit $106.95, High address option $8.95, Low address option $9.95. Custom Cabinet** with drilled and labelled plexiglass front panel **$24.95.** All metal Expansion Cabinet, painted and silk screened, with room for 5 S-100 boards and power supply **$57.00. NiCad Battery Memory Saver Kit $6.95.** All kits and options also completely assembled and tested.

Questdata, a software publication for 1802 computer users is available by subscription for $12.00 per 12 issues. Single issues $1.50. Issues 1-12 bound $16.50.

Tiny Basic Cassette **$10.00**, on ROM **$38.00**, original Elf kit board **$14.95. 1802 software;** Moews Video Graphics **$3.50.** Games and Music **$3.00**, Chip 8 Interpreter **$5.50.**

Super Expansion Board with Cassette Interface $89.95

This is truly an astounding value! This board has been designed to allow **you** to decide how you want it optioned. **The Super Expansion Board comes with 4K of low power RAM** fully addressable anywhere in 64K with built-in memory protect and a **cassette interface**. Provisions have been made for all other options on the same board and it fits neatly into the hardwood cabinet alongside the **Super Elf**. The board includes slots for up to 6K of **EPROM** (2708, 2758, 2716 or TI 2716) and is **fully socketed.** EPROM can be used for the monitor and Tiny Basic or other purposes.

A 1K Super ROM Monitor $19.95 is available as an on board option in 2708 EPROM which has been preprogrammed with a program loader/editor and error checking multi file cassette read/write software, (relocatable cassette file) another exclusive from Quest. It includes register save and readout, block move capability and video graphics driver with blinking cursor. Break points can be used with the register save feature to isolate program bugs quickly, then follow with single step. If you have the **Super Expansion Board** and **Super Monitor** the monitor is up and running at the push of a button.

Other on board options include **Parallel Input and Output Ports** with **full handshake.** They allow easy connection of an ASCII keyboard to the input port. **RS 232 and 20 ma Current Loop** for teletype or other device are on board and if you need more memory there are two **S-100** slots for static RAM or video boards. Also a 1K Super Monitor version 2 with video driver for full capability display with Tiny Basic and a video interface board. **Parallel I/O Ports $9.85, RS 232 $4.50, TTY 20 ma I/F $1.95, S-100 $4.50.** A **50 pin connector set** with ribbon cable is available at $15.25 for easy connection between the **Super Elf** and the **Super Expansion Board.**

Power Supply Kit for the complete system (see Multi-volt Power Supply).

Announcing Quest Super Basic— *SECOND GENERATION*

A new enhanced version of **Super Basic** now available. Quest was the first company worldwide to ship a full size Basic for 1802 Systems. A complete function **Super Basic** by **Ron Cenker** including floating point capability with scientific notation (number range $\pm .17E^{38}$), 32 bit integer ± 2 billion; multi dim arrays, string arrays; string manipulation; cassette I/O, save and load, basic, data and machine language programs; and over 75 statements, functions and operations.

Enhancements include increased speed, built-in provisions for Stringy, Floppy Disc, Printer Driver, I/O, user definable command library and statement renumbering.

Easily adaptable to most 1802 systems. Requires 16K RAM minimum for Basic and user programs. Source listing for both Serial and Parallel I/O included.

Super Basic on Cassette $40.00.

Gremlin Color Video Kit $69.95

32 x 16 alpha/numerics and graphics; up to 8 colors with 6847 chip; 1K RAM at E000. Plugs into Super Elf 44 pin bus. No high res. graphics. On board RF Modulator Kit **$4.95**

Elf II Adapter Kit $24.95

Plugs into Elf II providing Super Elf 44 and 50 pin plus S-100 bus expansion. (With Super Expansion). High and low address displays, state and mode LED's optional **$18.00.**

1802 16K Dynamic RAM Kit $149.00

Expandable to 32K. Hidden refresh w/clocks up to 4 MHz w/no wait states. Addl. 16K RAM **$63.00**

Super Elf 44 pin expansion board; 3 female and 1 male bus. Board plus 3 connectors **$22.95**

Tiny Basic Extended on Cassette	**$15.00**

(added commands include Stringy, Array, Cassette I/O etc.)

S-100 4-Slot Expansion	**$ 9.95**
Super Monitor VI.I Source Listing	**$15.00**

Super Color S-100 Video Kit $129.95

Expandable to 256 x 192 high resolution color graphics. 6847 with all display modes computer controlled. Memory mapped. 1K RAM expandable to 6K. S-100 bus 1802, 8080, 8085, Z80 etc.

Editor Assembler	**$25.00**

(Requires minimum of 4K for E/A plus user source)

1802 Tiny Basic Source listing	**$19.00**
Super Monitor V2.0/2.1 Source Listing	**$20.00**

PROM Eraser

assembled. 25 PROM capacity **$37.50** (with timer **$69.50**). 6 PROM capacity OSHA/UL version **$69.50** (with timer **$94.50**).

Z80 Microcomputer

16 bit I/O, 2 MHz clock, 2K RAM, ROM Breadboard space. Excellent for control. Bare Board **$28.50.** Full Kit **$99.00.** Monitor **$20.00.** Power Supply Kit **$35.00. Tiny Basic $30.00**

S-100 Computer Boards

8K Static Godbout Econo IIA Kit	145.00
16K Static Godbout Econo XIV Kit	285.00
24K Static Godbout Econo VIIA-24 Kit	435.00
32K Static Godbout Econo X-32 Kit	575.00
16K Dynamic RAM Kit	199.00
32K Dynamic RAM Kit	310.00
64K Dynamic RAM Kit	470.00
Video Interface Kit	**$135.00**

80 IC Update Master Manual $55.00

Comp. IC data selector, 2700 pg. master reference guide. Over 51,000 cross references. Free update service through 1980. Domestic postage $3.50.

Modem Kit $60.00

State of the art, orig., answer. No tuning necessary. 103 compatible 300 baud. Inexpensive acoustic coupler plans included.

LRC 7000 + Printer $389.00

40/20 column dot matrix impact, std. paper. Interface all personal computers.

64/40/32/20 version **$405.00.** Optional cables available.

LRC 7000 printer interface cable for Super Elf with software $26.00

NiCad Battery Fixer/Charger Kit

Opens shorted cells that won't hold a charge and then charges them up, all in one kit w/full parts and instructions. **$7.25**

Rockwell AIM 65 Computer

6502 based single board with full ASCII keyboard and 20 column thermal printer. 20 char. alphanumeric display, ROM monitor, fully expandable. **$375.00.** 4K version **$450.00.** 4K Assembler **$85.00.** 8K Basic Interpreter **$100.00.**

Special small power supply for AIM65 assem. in frame **$54.00.** Complete AIM65 in thin briefcase with power supply **$499.00.** Molded plastic enclosure to fit both AIM65 and power supply **$47.50.** Special Package Price: 4K AIM, 8K Basic, power supply, cabinet **$599.00**

AIM65/KIM/VIM/Super Elf 44 pin expansion board; 3 female and 1 male bus. Board plus 3 connectors **$22.95.**

60 Hz Crystal Time Base Kit $4.40

Converts digital clocks from AC line frequency to crystal time base. Outstanding accuracy.

Video Modulator Kit $8.95

Convert TV set into a high quality monitor w/o affecting usage. Comp. kit w/full instruc.

Multi-volt Computer Power Supply

8v 5 amp, ±18v .5 amp, 5v 1.5 amp, −5v .5 amp, 12v .5 amp, −12v option. ±5v, ±12v are regulated. Basic Kit **$29.95.** Kit with chassis and all hardware **$43.95.** Add $4.00 shipping. Kit of hardware **$14.00.** Woodgrain case **$10.00.** $1.50 shipping.

TERMS: $5.00 min. order U.S. Funds. Calif residents add 6% tax.
$10.00 min. order BankAmericard and Master Charge and COD. $1.00 insurance optional.
Shipping charges will be added on charge cards.

FREE: Send for your copy of our NEW 1980 QUEST CATALOG. Include 48¢ stamp.

apple II plus
With 48K of memory!
$1042.00
With the purchase of the APPLE II select from the below SPECIAL PRICING!
Base2 printer.. $599.00
Disk II w/cont.... 585.00
Disk II.............. 475.00
Ser. Printer Cd... 179.00
SupR'Mod......... 23.00
3way I/O Select.. 33.00
Video 100 12".... 119.00
Firmware Card. 149.00
UHF to RCA Cable. 5.00
apple II/apple II plus
With 60K of memory!
$1259.00
Switch from "APPLESOFT" to "INTEGER BASIC" and back again.
With Disc Operating System DOS 3.3 run PASCAL program without need for a PASCAL card.
$159
BASE 2 PRINTER TO APPLE COMPUTER
INTERFACE BOARD WITH CABLE
$124.95
apple clock/calendar
Seconds, minutes, hours, day-of-week month, date, & year.
On board batteries with one year life.
Uses MSM5832-crystal controlled.
California Computer Systems
S-100 MEMORY BOARD
16K STATIC RAM
$249.00
ASSEMBLED & TESTED
FULLY STATIC OPERATION
4K BANK ADDRESSABLE
EXTENDED MEMORY MGMT
MEETS IEEE PROPOSED
S-100 SIGNAL STANDARDS
4 MHZ OPERATION
California Computer Systems
74LS00 .26
74LS02 .26
74LS03 .26
74LS04 .26
74LS08 .28
74LS09 .26
74LS10 .26
74LS20 .26
74LS21 .28
74LS22 .26
74LS26 .49
74LS27 .26
74LS30 .28
74LS32 .32
74LS38 .32
74LS42 .65
74LS48 .78
74LS51 .25
74LS54 .35
74LS74 .38
74LS75 .60
74LS83 .44
74LS85 .95
74LS86 .95
74LS90 .69
74LS93 .69
74LS107 .45
74LS112 .38
74LS113 .48
74LS122 .48
74LS123 .95
74LS126 .69
74LS138 .69
74LS151 .44
74LS153 .44
74LS155 1.15
74LS158 .75
74LS160 .95
74LS161 .85
74LS162 .95
74LS163 1.60
74LS164 .65
74LS165 .65
74LS170 1.75
74LS174 .75
74LS175 .75
74LS190 .75
74LS193 .95
74LS195 .95
74LS196 .85
74LS221 1.40
74LS240 1.65
74LS241 1.65
74LS243 1.45
74LS244 1.45
74LS245 2.25
74LS253 .95
74LS257 .95
74LS258 .95
74LS259 2.85
74LS279 .44
74LS283 1.00
74LS293 1.85
74LS298 1.20
74LS366 .95
74LS367 .55
74LS368 .55
74LS373 1.39
74LS374 1.39
74LS386 .65
Global Specialties
Logic Probes
LP1 $37.49
LP2 $21.49
LP3 $58.49
$87.50
No "Glitches", Surges Or Interference
The MPD 117 is the low-cost solution to your power distribution problems. The MPD 117 has a high-performance EMI filter, built-in circuit breaker, two direct and six switched outlets, illuminated "on-off" switch for switched outlets, and is built with rugged UL approved components and housing in an all steel chassis with convenient mounting flanges.
MSM5832
MICROPROCESSOR REAL-TIME CLOCK/CALENDAR $7.45
GENERAL DESCRIPTION
The MSM5832 is a monolithic metal-gate CMOS integrated circuit that functions as a real time clock calendar for use in bus-oriented microprocessor applications. The on-chip 32 768 Hz crystal controlled oscillator time base is counted down to provide addressable 4-bit I/O data of SECONDS, MINUTES, HOURS, DAY-OF-WEEK, DATE, MONTH and YEAR. Data access is controlled by 4-bit address, chip select, read, write and hold inputs. Other functions include 12H/24H format selection, leap year identification and manual ± 30 second correction.
5712
Panel W-1¾ H-5¼
Shelf W-3½ H-5¼
5722
DTE A
DCE
DTE B
Connector Layout
$145.00 $155.00
Description
The RS-232C Compatible Digital Transfer Switch is designed to switch modems between front end processors. All 24 pins of the connector are switched, with Pin 1 wired to ground.
T-bar INCORPORATED
EPROMS
2708 1K×8 $ 5.95 8 for $40
2716 2K×8 10.95 8 for $80
2732 4K×8 39.95
SINGLE 5 VOLT FOR 2716 & 2732
TRS 80
16K Memory Add On
$43.95
KIT CONTAINS DIP SWITCHES AND DETAILED INSTRUCTIONS
SN7400N .20 SN7482N 1.05
SN7402N .22 SN7492N .50
SN7404N .22 SN7493N .48
SN7408N .24 SN7495N .60
SN7410N .22 SN7496N .70
SN7412N .28 SN74122N .39
SN7413N .35 SN74136N .95
SN7414N .49 SN74141N .69
SN7416N .29 SN74151N .65
SN7417N .29 SN74153N .65
SN7423N .28 SN74154N 1.25
SN7425N .25 SN74155N .80
SN7430N .23 SN74157N .69
SN7437N .29 SN74160N .95
SN7438N .24 SN74161N .65
SN7440N .22 SN74163N .85
SN7442N .57 SN74164N .87
SN7443N .78 SN74165N .87
SN7445N .78 SN74174N .95
SN7451N .20 SN74175N .69
SN7454N .20 SN74180N .75
SN7474N .20 SN74181N 1.15
SN7475N .32 SN74393N 1.69
video 100
$125
Leedex Corp.
12" BLACK & WHITE LOW COST VIDEO MONITOR
MODEL 800B IMPACT PRINTER
Dot resolution graphics in six densities
1920 character buffer
Fully adjustable tractors to 9-1/2
$649.00
base 2 inc.
TMS2716 E-PROM
REQUIRES THREE $8.50 POWER SUPPS.
CONCORD
COMPUTER COMPONENTS
297
1971 SOUTH STATE COLLEGE ANAHEIM, CA. 92806
VISA MASTERCHARGE
CHECK OR M O
NO COD
(714) 937-0637
MINIMUM ORDER $10.00
ADD $1.50 FOR FRT
CAL RES ADD 6%
We stock and sell over 12,000 types of semiconductors
FRT. CHARGES
am't frt. chg.
$10-49 $2.00
50-99 4.00
100-249 7.00
250-499 8.00
500-999 10.00
1000 up inquire
VOLTAGE REGULATORS
NEGATIVE 7905 5V 7908 8V 7915 15V 7918 18V
POSITIVE 7805 5V 7806 6V 7808 8V 7812 12V
95¢
MC1330A1P $1.60
MC1350P 1.15
MC1358P 1.50
LM380N 1.75
NE565N 1.25
MC1458P .50
555 TIMER 27¢
APPLE EXPANSION KIT
16K Memory Add-On $39.00
MEMORY ADD-ON KIT INCLUDES INSTRUCTIONS
S-100 16K ADD-ON BARE BOARD WITH DOCUMENTATION AND DETAILED INSTRUCTION BOOK
$28.95
8038C
VCO Waveform Gen w/sine $2.65
2114L
1024×4 Static RAM
300nsec $3.99
200nsec 4.75
SCR's & Triacs
10 Amp. 400 Volts Isolated Tab
TO-220 Pkg
IS410 95¢
IT410 95¢
SOROC TECHNOLOGY, INC
IQ120 $689
CSC
$17.95
LOGIC PROBE KIT
COMPLETE 8085A BASED MICRO-COMPUTER AND MICROPROCESSOR TRAINING UNIT
YOU GET:
Fully assembled and tested 8085A Microcomputer with 1K RAM, 1K PROM, & 1K EPROM Programmable I/O, Keyboard unit, CPU card, 20 ma Asynchronous serial port, Display and Operating system
8085A Cookbook 362 pages takes you from basic concepts to actual design
8080/8085A Software Design Book I with detailed examination of all 244 instructions. 336 pages include sample programs using all basic instructions and typical assembly language
8085A COOKBOOK
PACCOM $295.00

7400

SN7400N	.25
SN7401N	.20
SN7402N	.25
SN7403N	.25
SN7404N	.25
SN7405N	.29
SN7406N	.35
SN7407N	.35
SN7408N	.29
SN7409N	.29
SN7410N	.25
SN7411N	.29
SN7412N	.35
SN7413N	.40
SN7414N	.69
SN7416N	.29
SN7417N	.29
SN7420N	.25
SN7421N	.29
SN7422N	.45
SN7423N	.29
SN7425N	.29
SN7426N	.29
SN7427N	.25
SN7429N	.39
SN7430N	.25
SN7432N	.29
SN7437N	.25
SN7438N	.40
SN7439N	.25
SN7440N	.20
SN7441N	.89
SN7442N	.59
SN7443N	1.10
SN7444N	1.10
SN7445N	.89
SN7446N	.79
SN7447N	.69
SN7448N	.79
SN7450N	.20
SN7451N	.20
SN7453N	.20
SN7454N	.20
SN7459A	.25
SN7460N	.20
SN7470N	.29

SN7472N	.29
SN7473N	.35
SN7474N	.35
SN7475N	.49
SN7476N	.35
SN7479N	5.00
SN7480N	.50
SN7482N	.99
SN7483N	.69
SN7485N	.89
SN7486N	.35
SN7489N	1.75
SN7490N	.49
SN7491N	.59
SN7492N	.45
SN7493N	.45
SN7494N	.69
SN7495N	.69
SN7496N	.69
SN7497N	3.00
SN74100N	1.49
SN74107N	.35
SN74109N	.39
SN74116N	1.95
SN74121N	.39
SN74122N	.55
SN74123N	.59
SN74125N	.49
SN74126N	.49
SN74132N	.75
SN74136N	.75
SN74141N	.99
SN74142N	3.25
SN74143N	3.49
SN74144N	3.49
SN74145N	.79
SN74147N	1.95
SN74148N	1.29
SN74150N	1.25
SN74151N	.69
SN74152N	.69
SN74153N	.75
SN74154N	1.25
SN74155N	.79

SN74156N	.79
SN74157N	.69
SN74160N	.89
SN74161N	.89
SN74162N	.89
SN74163N	.89
SN74164N	.89
SN74165N	.89
SN74166N	1.25
SN74167N	2.79
SN74170N	1.95
SN74172N	4.95
SN74173N	1.39
SN74174N	.99
SN74175N	.89
SN74176N	.79
SN74177N	.79
SN74179N	1.49
SN74180N	.79
SN74181N	2.25
SN74182N	.79
SN74184N	2.49
SN74185N	2.49
SN74190N	1.25
SN74191N	1.25
SN74192N	.89
SN74193N	.89
SN74194N	.89
SN74195N	.89
SN74196N	.89
SN74197N	.89
SN74198N	1.49
SN74199N	1.49
SN74221N	1.25
SN74251N	.99
SN74276N	1.95
SN74279N	.79
SN74283N	1.49
SN74284N	3.95
SN74285N	3.95
SN74365N	.69
SN74366N	.69
SN74367N	.69
SN74368N	.69
SN74390N	1.49
SN74393N	1.49

74LS

74LS00	.29
74LS01	.29
74LS02	.29
74LS03	.29
74LS04	.35
74LS05	.29
74LS08	.29
74LS09	.35
74LS10	.29
74LS11	.75
74LS12	.35
74LS13	.59
74LS14	.99
74LS15	.35
74LS20	.29
74LS21	.35
74LS22	.35
74LS26	.35
74LS27	.35
74LS28	.35
74LS30	.29
74LS32	.35
74LS33	.59
74LS37	.45
74LS38	.49
74LS40	.35
74LS42	.89
74LS47	.89
74LS48	1.15
74LS49	1.15
74LS51	.29
74LS54	.29
74LS55	.29
74LS73	.45
74LS74	.45
74LS75	.59
74LS76	.45
74LS78	.49
74LS83	.89
74LS85	1.25
74LS86	.45
74LS90	.59

74LS92	.75
74LS93	.75
74LS95	.99
74LS96	1.15
74LS107	.45
74LS109	.45
74LS112	.45
74LS113	.49
74LS114	.49
74LS122	.89
74LS123	1.25
74LS125	.89
74LS126	.55
74LS132	.99
74LS133	.89
74LS136	.49
74LS138	.89
74LS139	.89
74LS151	.89
74LS153	.89
74LS154	1.75
74LS155	1.19
74LS156	1.19
74LS157	.89
74LS158	.89
74LS160	1.15
74LS161	1.15
74LS162	1.15
74LS163	1.15
74LS164	1.25
74LS165	1.25
74LS168	1.19
74LS169	1.19
74LS170	2.49
74LS173	1.39
74LS174	.99
74LS175	.99
74LS181	2.95
74LS190	1.25
74LS191	1.25

74LS192	1.15
74LS193	1.15
74LS194	1.15
74LS195	1.15
74LS197	1.15
74LS221	1.19
74LS240	1.95
74LS241	1.95
74LS242	1.95
74LS243	1.95
74LS244	1.95
74LS245	2.95
74LS247	1.19
74LS248	1.19
74LS249	1.19
74LS251	1.49
74LS253	.99
74LS257	.89
74LS258	.99
74LS260	.69
74LS266	.69
74LS273	1.95
74LS279	.75
74LS283	1.09
74LS290	.99
74LS293	.99
74LS298	1.25
74LS352	1.29
74LS353	1.29
74LS365	.75
74LS366	.75
74LS367	.75
74LS368	.75
74LS373	1.95
74LS374	1.95
74LS375	.89
74LS386	.69
74LS393	2.49
74LS399	2.49
74LS670	2.49
81LS95	1.95
81LS97	1.95

74S

74S00	.50
74S02	.50
74S03	.50
74S04	.55
74S05	.55
74S08	.50
74S09	.50
74S10	.50
74S11	.50
74S15	.50
74S20	.50
74S22	.50
74S30	.50
74S32	.55
74S40	.55
74S51	.50
74S64	.50
74S65	.50
74S74	.79
74S86	.79
74S112	.79
74S113	.79
74S114	.79

74S133	.55
74S134	.69
74S135	1.19
74S136	1.75
74S138	1.35
74S139	1.35
74S140	1.15
74S151	1.35
74S153	1.35
74S157	1.35
74S158	1.35
74S174	1.59
74S175	1.59
74S188	4.95
74S194	1.95
74S195	1.95
74S196	3.95
74S240	2.95
74S241	2.95
74S243	3.25

74S244	3.25
74S251	1.45
74S253	1.45
74S257	1.35
74S258	1.35
74S260	.79
74S280	2.95
74S287*	4.95
74S288*	4.95
74S373	3.49
74S374	3.49
74S387*	5.95
74S471*	19.95
74S472*	19.95
74S473*	19.95
74S474*	21.95
74S475*	21.95
74S570*	7.95
74S571*	7.95
74S572*	19.95
74S573*	19.95
74S940	3.15
74S941	3.15

* LIMITED AVAILABILITY ON THESE PROMS

CA—LINEAR

CA3013H	2.15
CA3023H	3.25
CA3039H	1.35
CA3046N	1.30
CA3059N	3.25
CA3060N	3.25
CA3080H	1.25
CA3081N	2.00
CA3082N	2.00
CA3083N	1.60
CA3086N	.85
CA3089N	3.75
CA3096N	3.95
CA3130H	1.39
CA3140H	1.25
CA3160H	1.25
CA3401N	.59
CA3600N	3.50

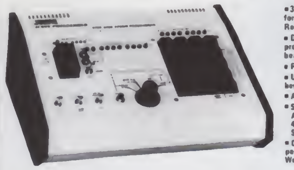
CD—CMOS

CD4000	.39
CD4001	.39
CD4002	.39
CD4006	1.19
CD4007	.25
CD4009	.49
CD4010	.49
CD4011	.39
CD4012	.25
CD4013	.49
CD4014	1.39
CD4015	1.19
CD4016	.59
CD4017	1.19
CD4018	.99
CD4019	.49
CD4020	1.19
CD4021	1.39
CD4022	1.19
CD4023	.29
CD4024	.79
CD4025	.23
CD4026	2.95
CD4027	.69
CD4028	.89
CD4029	1.49
CD4030	.49
CD4035	.99
CD4040	1.49

CD4041	1.49
CD4042	.99
CD4043	.89
CD4044	.89
CD4046	1.79
CD4047	2.50
CD4048	1.35
CD4049	.69
CD4050	.69
CD4051	1.19
CD4052	1.19
CD4053	1.19
CD4056	2.95
CD4059	9.95
CD4060	1.49
CD4066	.79
CD4068	.39
CD4069	.45
CD4070	.55
CD4071	.49
CD4072	.49
CD4073	.39
CD4075	.39
CD4076	1.39
CD4081	.39

CD4082	.39
CD4093	.99
CD4096	2.49
CD4506	.75
CD4507	.99
CD4508	3.95
CD4510	1.39
CD4511	1.29
CD4514	3.95
CD4515	2.95
CD4516	1.49
CD4518	1.79
CD4519	.89
CD4520	1.29
CD4526	1.79
CD4528	1.79
CD4529	1.95
CD4543	2.79
CD4562	11.95
CD4566	2.79
CD4583	2.49
CD4723	1.95
CD4724	1.95
MC14409	14.95
MC14410	14.95
MC14411	14.95
MC14412	11.95
MC14419	4.95
MC14433	13.95

JE608 PROGRAMMER
2708 EPROM PROGRAMMER



- 3 separate Display Registers: 8 LED's for Hex Key entries, 10 LED's (2^0-2^9) for Address Register and 8 LED's for Data Memory Register. The Data Memory Register displays the content of the RAMs from the EPROM Chip.
- Development of microprocessor systems by means of a ribbon cable from the programmer panel test socket to the EPROM socket on the microprocessor board.
- Rapid checking verification of programmed data changes.
- User may move data from a master to RAM's or write into RAM's with keyboard entries.
- Allows manual stepping manipulation (up and down) at any address location.
- Stand alone EPROM Programmer consisting of: A 19-key Hexadecimal Keyboard assembly, Programmer Board assembly with 4 power supplies and a LED/Test Socket Panel Board assembly. The Test Socket is zero force insertion type. Power requirements 115VAC, 60Hz, 6W.
- Compact desk top enclosure: Color-coordinated designer's case with light tan panels and molded end pieces in mocha brown. Size: 3¾"H x 11"W x 8¾"D. Weight: 5 lbs.

The JE608 EPROM Programmer is a completely self contained unit which is independent of computer control and requires no additional systems for its operations. The EPROM can be programmed from the Hexadecimal Keyboard or from a pre-programmed EPROM. The JE608 Programmer can emulate a programmed EPROM by the use of its internal RAM circuits. This will allow the user to test or pretest a program, for a system, prior to programming a chip. Any changes in the program can be entered directly into the memory circuits with the Hexadecimal Keyboard so that rewriting the entire program will not be necessary. The JE608 Programmer contains a Programmer/Board with 25 IC's and including power supplies of -5V, +5V, +12V and +26V. The Hexadecimal Keyboard and LED/Test Socket Panel Board are separate assemblies within the system.

JE608K KIT $399.95
JE608A Assembled and tested $499.95

DISCRETE LEDS

XC556R	.200"	red	5/$1
XC556G	.200"	green	4/$1
XC556Y	.200"	yellow	4/$1
XC556C	.200"	clear	4/$1
XC22R	.200"	red	5/$1
XC22G	.200"	green	4/$1
XC22Y	.200"	yellow	4/$1
MV10B	.170"	red	4/$1
MV50	.085"	red	6/$1
XC209R	.125"	red	5/$1
XC209G	.125"	green	4/$1
XC209Y	.125"	yellow	4/$1
XC526R	.185"	red	5/$1
XC526G	.185"	green	4/$1
XC526Y	.185"	yellow	4/$1
XC526C	.185"	clear	4/$1
XC111R	.190"	red	5/$1
XC111G	.190"	green	4/$1
XC111Y	.190"	yellow	4/$1
XC111C	.190"	clear	4/$1

INFRA-RED LED
¼"x¼"x1/16" flat
IRL — 5/$1

DISPLAY LEDS

C.A. — Common Anode C.C. — Common Cathode

Type	Polarity	Ht	Price
MAN 1	C.A.—red	.270	2.95
MAN 2	5x7 D.M.—red	.300	4.95
MAN 3	C.C.—red	.125	.25
MAN 52	C.A.—green	.300	1.25
MAN 54	C.C.—green	.300	1.25
MAN 71	C.A.—red	.300	.75
MAN 72	C.A.—red	.300	.75
MAN 74	C.C.—red	.300	1.25
MAN 82	C.A.—yellow	.300	.49
MAN 84	C.C.—yellow	.300	.99
MAN 3620	C.A.—orange	.300	.49
MAN 3630	C.A.—orange ± 1	.300	.99
MAN 3640	C.C.—orange	.300	.99
MAN 4610	C.A.—orange	.400	.99
MAN 6610	C.A.—orange—DD	.560	.99
MAN 6630	C.A.—orange ± 1	.560	.99
MAN 6640	C.C.—orange—DD	.560	.99
MAN 6650	C.C.—orange ± 1	.560	.99
MAN 6660	C.A.—orange	.560	.99
MAN 6710	C.A.—red—DD	.560	.99
MAN 6750	C.C.—red ± 1	.560	.99
MAN 6780	C.C.—red	.560	.99
DL704	C.C.—red	.300	1.25
DL707	C.A.—red	.300	1.25
DL728	C.C.—red	.500	1.49
DL741	C.A.—red	.600	1.25
DL746	C.A.—red ± 1	.630	1.49
DL747	C.A.—red	.600	1.49
DL750	C.C.—red	.600	1.49
DL33B	C.C.—red	.110	.35
FND358	C.C. ± 1	.357	.99
FND359	C.C.	.357	.75
FND503	C.C. (FND500)	.500	.99
FND507	C.A. (FND510)	.500	.99
HDSP-3401	C.A.—red	.800	1.50
HDSP-3403	C.C.—red	.800	1.50
5082-7613	C.C., R.H.D.—red	.300	1.25
5082-7620	C.A., L.H.D.—yel.	.300	1.25
5082-7623	C.C., R.H.D.—yel.	.300	1.25
5082-7730	C.A., L.H.D.—red	.300	.99
5082-7731	C.A., R.H.D.—red	.300	1.25
5082-7750	C.A., L.H.D.—red	.430	1.75
5082-7751	C.A., R.H.D.—red	.430	1.25
5082-7760	C.C., R.H.D.—red	.430	1.75
5082-7300	4x7 sgl. dig. RHD	.600	22.00
5082-7302	4x7 sgl. dig. LHD	.600	22.00
5082-7304	Overnge. char. (±1)	.600	19.95
4N28	Photo Xsistor Opto-Isol.		.99
LIT-1	Photo Xsistor Opto-Isol.		.69
MOC3010	Optically Isol. Triac Driver		1.25

POTENTIOMETERS

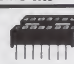
$2.95 each
2 Watt @ 70°C ± 10%
7/8" Slotted Shaft
Linear Taper
Meets Mil-R-94

Part No.		
CMU 1021	RV4NAY SD - 102A	1K
CMU 5021	RV4NAY SD - 502A	5K
CMU 1031	RV4NAY SD - 103A	10K
CMU 2531	RV4NAY SD - 253A	25K
CMU 5031	RV4NAY SD - 503A	50K
CMU 1041	RV4NAY SD - 104A	100K
CMU 1052	RV4NAY SD - 105A	1Meg

$1.35 each
3/4 Watt @ 70°C
15 turn pot. Linear taper. Printed circuit mount.

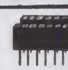
Part No.	Part No.	Part No.
830P- 50ohm	830P- 5K	830P-100K
830P-100ohm	830P-10K	830P-200K
830P-500ohm	830P-20K	830P-500K
830P-1K	830P-50K	830P-1Meg
830P-2K		

LOW PROFILE (TIN) SOCKETS



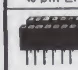
	1-24	25-49	50-100
8 pin LP	.17	.16	.15
14 pin LP	.20	.19	.18
16 pin LP	.22	.21	.20
18 pin LP	.29	.28	.27
20 pin LP	.34	.32	.30
22 pin LP	.37	.36	.35
24 pin LP	.38	.37	.36
28 pin LP	.45	.44	.43
36 pin LP	.60	.59	.58
40 pin LP	.63	.62	.61

SOLDERTAIL STANDARD (TIN)



	1-24	25-49	50-100
14 pin ST	.27	.25	.24
16 pin ST	.30	.27	.25
18 pin ST	.35	.32	.30
24 pin ST	.49	.45	.42
28 pin ST	.99	.90	.81
36 pin ST	1.39	1.26	1.15
40 pin ST	1.59	1.45	1.30

SOLDERTAIL (GOLD) STANDARD



	1-24	25-49	50-100
8 pin SG	.39	.35	.31
14 pin SG	.49	.45	.41
16 pin SG	.54	.49	.44
18 pin SG	.59	.53	.48
24 pin SG	.79	.75	.69
28 pin SG	1.10	1.00	.90
36 pin SG	1.65	1.40	1.26
40 pin SG	1.75	1.59	1.45

WIRE WRAP SOCKETS (GOLD) LEVEL #3

	1-24	25-49	50-100
8 pin WW	.59	.54	.49
10 pin WW	.69	.63	.58
14 pin WW	.79	.73	.67
16 pin WW	.85	.77	.70
18 pin WW	.99	.90	.81
20 pin WW	1.19	1.08	.99
22 pin WW	1.49	1.35	1.23
24 pin WW	1.39	1.26	1.14
28 pin WW	1.69	1.53	1.38
36 pin WW	2.19	1.99	1.79
40 pin WW	2.29	2.09	1.89

1/4 WATT RESISTOR ASSORTMENTS – 5%

ASST. 1	5ea.	10 Ohm 12 Ohm 15 Ohm 18 Ohm 22 Ohm 27 Ohm 33 Ohm 39 Ohm 47 Ohm 56 Ohm	50pcs.	$1.95
ASST. 2	5ea.	68 Ohm 82 Ohm 100 Ohm 120 Ohm 150 Ohm 180 Ohm 220 Ohm 270 Ohm 330 Ohm 390 Ohm	50pcs.	$1.95
ASST. 3	5ea.	470 Ohm 560 Ohm 680 Ohm 820 Ohm 1K 1.2K 1.5K 1.8K 2.2K 2.7K	50pcs.	$1.95
ASST. 4	5ea.	3.3K 3.9K 4.7K 5.6K 6.8K 8.2K 10K 12K 15K 18K	50pcs.	$1.95
ASST. 5	5ea.	22K 27K 33K 39K 47K 56K 68K 82K 100K 120K	50pcs.	$1.95
ASST. 6	5ea.	150K 180K 220K 270K 330K 390K 470K 560K 680K 820K	50pcs.	$1.95
ASST. 7	5ea.	1M 1.2M 1.5M 1.8M 2.2M 2.7M 3.3M 3.9M 4.7M 5.6M	50pcs.	$1.95
ASST. 8R		Includes Resistor Assts. 1-7 (350 pcs.)		$10.95 ea.

$10.00 Min. Order — U.S. Funds Only
Calif. Residents Add 6% Sales Tax
Postage — Add 5% plus $1 Insurance

✓41

Spec Sheets — 25¢
Send 41¢ Postage for your
FREE 1981 JAMECO CATALOG

NEW 1981 CATALOG

Jameco ELECTRONICS

PHONE ORDERS WELCOME
(415) 592-8097

MAIL ORDER ELECTRONICS — WORLDWIDE
1355 SHOREWAY ROAD, BELMONT, CA 94002
PRICES SUBJECT TO CHANGE

1/81

INTERSIL

Part No.	Function	Price
7045IPI	CMOS Precision Timer	14.95
7045EV/Kit*	Stopwatch Chip, XTL	22.95
7106CPL	3½ Digit A/D (LCD Drive)	16.95
7106EV/Kit*	IC, Circuit Board, Display	34.95
7107CPL	3½ Digit A/D (LED Drive)	15.95
7107EV/Kit*	IC, Circuit Board, Display	28.95
7116CPL	3½ Digit A/D LCD Dis. HLD.	18.95
7117CPL	3½ Digit A/D LED Dis. HLD.	17.95
7201DR	Low Battery Volt Indicator	2.25
7205IPG	CMOS LED Stopwatch/Timer	12.95
7205EV/Kit*	Stopwatch Chip, XTL	19.95
7206CJPE	Tone Generator	5.15
7206CEV/Kit*	Tone Generator Chip, XTL	9.95
7207AIPD	Oscillator Controller	6.50
7207AEV/Kit*	Freq. Counter Chip, XTL	11.10
7208IPI	Seven Decade Counter	17.95
7209IPA	Clock Generator	3.95
7215IPG	4 Func. CMOS Stopwatch CKT	13.95
7215EV/Kit*	4 Func. Stopwatch Chip, XTL	19.95
7216AIJI	8-Digit Univ. Counter C.A.	32.00
7216CIJI	8-Digit Freq. Counter C.A.	26.95
7216DIPI	8-Digit Freq. Counter C.C.	21.95
7217IJI	4-Digit LED Up/Down Counter	12.95
7218CIJI	8-Digit Univ. LED Drive	10.95
7224IPL	LCD 4½ Digit Up Counter DRI	11.25
7226AIJL	8-Digit Univ. Counter	31.95
7226AEV/Kit*	5 Function Counter Chip, XTL	74.95
7240IJE	CMOS Bin Prog. Timer/Counter	4.95
7242IJA	CMOS Divide-by-256 RC Timer	2.05
7250IJE	CMOS BCD Prog. Timer/Counter	6.00
7260IJE	CMOS BCD Prog. Timer/Counter	5.25
7555IPA	CMOS 555 Timer (8 pin)	1.45
7556IPD	CMOS 556 Timer (14 pin)	2.20
7611BCPA	CMOS Op Amp Comparator 5MV	2.25
7612BCPA	CMOS Op Amp Ext. Cmvr. 5MV	2.95
7621BCPA	CMOS Dual Op Amp Comp. 5MV	3.95
7631CCPE	CMOS Tri Op Amp Comp. 10MV	5.35
7641CCPD	CMOS Quad Op Amp Comp. 10MV	7.50
7642CCPD	CMOS Quad Op Amp Comp. 10MV	7.50
7660CPA	Voltage Converter	2.95
8069CCQ	50ppm Band—GAP Volt Ref. Diode	2.50
8211CPA	Volt Ref/Indicator	2.50
8212CPA	Volt Ref/Indicator	2.50

* INTERSIL'S EVALUATION KITS

74C

74C00	.39
74C02	.39
74C04	.39
74C08	.39
74C10	.39
74C14	.75
74C20	.39
74C30	.39
74C42	1.39
74C48	1.95
74C73	.79
74C74	.79
74C85	1.95
74C86	.99
74C89	6.95
74C90	1.29
74C93	1.29
74C95	1.59

74C106	.75
74C107	1.89
74C151	2.95
74C154	3.95
74C157	2.25
74C160	1.69
74C161	1.60
74C162	1.49
74C163	1.69
74C164	1.59
74C173	1.39
74C174	1.39
74C175	1.39
74C192	1.69
74C193	1.69

74C195	1.59
74C221	1.95
74C240	2.25
74C244	2.25
74C373	2.49
74C374	2.59
74C901	.89
74C903	1.15
74C911	10.95
74C912	10.95
74C915	1.69
74C917	10.95
74C922	5.49
74C923	5.75
74C925	7.50
74C926	7.50
80C95	.79
80C97	.79

LINEAR

LH0002CN	6.85
LM10CLH	4.50
LM11CLH	4.75
LH0070-OH	6.05
TL071CP	.79
TL072CP	1.39
TL074CN	2.49
LH0082CD	35.80
TL082CP	1.19
TL084CN	2.19
LH0094CD	36.80
LM300H	.99
LM301CN	.35
LM302H	1.95
LM304H	1.95
LM305H	.99
LM307CN	.45
LM308CN	1.00
LM309H	1.95
LM309K	1.25
LM310CN	1.75
LM311H	.90
LM312H	2.49
LM317MP	1.15
LM317T	1.75
LM317K	3.95
LM318CN	1.95
LM319N	1.95
LM320K-5	1.35
LM320K-12	1.35
LM320K-15	1.35
LM320T-5	1.25
LM320T-12	1.25
LM320T-15	1.25
LM323K-5	5.95
LM324N	.99
LM329DZ	.65
LM331N	3.95
LM334Z	1.30
LM335Z	1.40
LM336Z	1.75
LM337T	1.95
LM337MP	1.15
LM338K	6.95
LM339N	.99
LM340K-5	1.35
LM340K-12	1.35
LM340K-15	1.35

LM340T-5	1.25
LM340T-12	1.25
LM340T-15	1.25
LM341P-5	.75
LM341P-12	.75
LM341P-15	.75
LM342P-5	.69
LM342P-12	.69
LM342P-15	.69
LM348N	1.25
LM350K	5.75
LF351N	.60
LF353N	1.00
LF355N	1.10
LF356N	1.10
LM358N	1.00
LM359N	1.79
LM370N	4.49
LM373N	3.25
LM377N	2.95
LM380N	1.25
LM381N	1.95
LM382N	1.79
LM384N	1.95
LM386N-3	1.29
LM387N	1.45
LM389N	1.35
LM392N	.69
LF398N	4.00
LM399H	5.00
TL494CN	4.49
TL496CP	1.75
NE510A	6.00
NE529A	4.95
NE531H	3.95
NE536H	6.00
NE540H	6.00
NE544N	4.95
NE550A	1.30
NE555V	.39
LM556N	.99
NE564N	3.95
LM565N	1.25
LM566CN	1.95
LM567V	1.25

NE570N	4.95
LM702H	.79
LM703CN	.89
LM709N	.29
LM710N	.79
LM711N	.79
LM723N	.69
LM733N	1.00
LM739N	1.19
LM741CN	.35
MC1741SCG	3.00
LM747N	.79
LM748N	.59
LM1014N	2.75
LM1310N	1.95
LM1458CN	.59
LM1488N	1.25
LM1489N	1.25
LM1496N	1.95
LM1556V	1.75
LM1800N	2.95
LM1877N-9	3.25
LM1889N	3.20
LM1896N	1.75
LM2002T	1.49
LM2877P	2.05
LM2878P	2.25
LM2896P-1	2.25
LM3189N	2.95
LM3900N	.69
LM3905CN	1.25
LM3909N	1.15
LM3914N	3.95
LM3915N	3.95
LM3916N	3.95
RC4136N	1.25
RC4151NB	3.95
RC4194TK	5.95
RC4195TK	5.49
KB4428	4.25
KB4429	5.95
LM4500A	3.25
ICL8038B	4.95
LM13080N	1.29
LM13600N	1.49
75138N	1.95
75450N	.89
75451CN	.39

CAPACITOR CORNER

50 VOLT CERAMIC DISC CAPACITORS

Value	1-9	10-99	100+	Value	1-9	10-99	100+
10 pf	.08	.06	.05	.001μF	.08	.06	.05
22 pf	.08	.06	.05	.0047μF	.08	.06	.05
47 pf	.08	.06	.05	.01μF	.08	.06	.05
100 pf	.08	.06	.05	.022μF	.09	.07	.06
220 pf	.08	.06	.05	.047μF	.09	.07	.06
470 pf	.08	.06	.05	.1μF	.15	.12	.10

100 VOLT MYLAR FILM CAPACITORS

.001mf	.12	.10	.07	.022mf	.13	.11	.08
.0022mf	.12	.10	.07	.047mf	.21	.17	.13
.0047mf	.12	.10	.07	.1mf	.27	.23	.17
,01mf	.12	.10	.07	.22mf	.33	.27	.22

+20% DIPPED TANTALUMS (Solid) CAPACITORS

.1/35V	.39	.34	.29	1.5/35V	.41	.37	.29
.15/35V	.39	.34	.29	2.2/35V	.51	.45	.34
.22/35V	.39	.34	.29	3.3/25V	.53	.47	.37
.33/35V	.39	.34	.29	4.7/25V	.63	.56	.45
.47/35V	.39	.34	.29	6.8/25V	.79	.69	.55
.68/35V	.39	.34	.29	15/25V	1.39	1.25	.95
1.0/35V	.39	.34	.29	22/6V	.79	.69	.55

MINI. ALUMINUM ELECTROLYTIC CAPACITORS

Axial Lead				Radial Lead			
.47/50V	.16	.14	.10	.47/25V	.15	.13	.12
1.0/50V	.19	.16	.12	.47/50V	.16	.14	.13
3.3/50V	.17	.15	.11	1.0/16V	.15	.13	.12
4.7/25V	.18	.15	.11	1.0/25V	.16	.14	.13
10/25V	.18	.15	.11	1.0/50V	.17	.15	.14
10/50V	.19	.16	.12	4.7/16V	.15	.13	.12
22/25V	.19	.16	.12	4.7/25V	.16	.14	.13
22/50V	.24	.20	.18	4.7/50V	.17	.15	.14
47/25V	.25	.21	.19	10/16V	.15	.13	.12
47/50V	.29	.25	.23	10/25V	.16	.14	.13
100/25V	.28	.24	.22	10/50V	.17	.15	.14
100/50V	.41	.37	.34	47/50V	.25	.21	.19
220/25V	.39	.34	.33	100/16V	.21	.17	.14
220/50V	.49	.45	.41	100/25V	.25	.23	.21
470/25V	.54	.49	.45	100/50V	.37	.34	.31
1000/16V	.79	.69	.61	220/16V	.25	.21	.19
2200/16V	.89	.79	.69	470/25V	.35	.31	.27

kilobaud Peterborough NH 03458

MICROCOMPUTING™ LIST OF ADVERTISERS

***This advertiser prefers to be contacted directly.**

kilobaud MICROCOMPUTING

For further information from our advertisers, please use the Reader Service card

READER SERVICE

Please help us to bring you a better magazine—by answering these questions:

I. I am interested enough in a general-interest article to see it in lieu of a technical article.
☐ 1 Yes
☐ 2 No

II. The following types of general interest articles appeal to me most (check two only).
☐ A. Humor
☐ B. Interview with users/industry leaders
☐ C. Historical
☐ D. Essay
☐ E. Profile of a business

III. As a reader of Microcomputing, I consider myself a
☐ 1 Beginner
☐ 2 Intermediate
☐ 3 Expert

IV. The articles in Microcomputing are
☐ A. Too simple
☐ B. Too complex
☐ C. Just right

V. I want to see more industry news in Microcomputing
☐ 1 Yes
☐ 2. No

VI. The following major computer publication gives me the most usable information (check only one)
☐ A. Kilobaud Microcomputing
☐ B. 80 Microcomputing
☐ C. Byte
☐ D. Interface Age
☐ E. Personal Computing
☐ F. Creative Computing
☐ G. Dr. Dobb's Journal

VII. I want to see more regular departments in Microcomputing
☐ 1 Yes
☐ 2 No

VIII. I bought this magazine on the newsstand because (answer only if question applies)
☐ A. I was looking specifically for Microcomputing
☐ B. The cover caught my eye

IX. I want to see more of the following kinds of articles in Microcomputing (check two only)
☐ 1 Business
☐ 2 Construction
☐ 3 Education
☐ 4 Game
☐ 5 Home application
☐ 6 Human interest
☐ 7 Programming technique
☐ 8 Review
☐ 9 Scientific

X. Is the advertising content of the magazine a primary reason for your buying Kilobaud?
☐ A. Yes
☐ B. No

XI. What type of computer do you own? (Check all that apply)
☐ 1 Apple
☐ 2 PET
☐ 3 TRS-80
☐ 4 Atari
☐ 5 Heath
☐ 6 TI
☐ 7 Other

XII. If you are not a subscriber, please circle 500.

Reader Service: Return this card to receive full information on the products advertised in this issue. Refer to the ad. You will find numbers near the logo of each advertiser. Each represents the advertiser's individual Reader Service number. Circle the corresponding numbers on one of the cards on this page, include your name, address & zip, and drop in a mailbox. In 4-6 weeks you'll hear from the advertiser directly.

This card valid until March 31, 1981.

1	6	11	16	21	126	131	136	141	146	251	256	261	266	271	376	381	386	391	396
2	7	12	17	22	127	132	137	142	147	252	257	262	267	272	377	382	387	392	397
3	8	13	18	23	128	133	138	143	148	253	258	263	268	273	378	383	388	393	398
4	9	14	19	24	129	134	139	144	149	254	259	264	269	274	379	384	389	394	399
5	10	15	20	25	130	135	140	145	150	255	260	265	270	275	380	385	390	395	400
26	31	36	41	46	151	156	161	166	171	276	281	286	291	296	401	406	411	416	421
27	32	37	42	47	152	157	162	167	172	277	282	287	292	297	402	407	412	417	422
28	33	38	43	48	153	158	163	168	173	278	283	288	293	298	403	408	413	418	423
29	34	39	44	49	154	159	164	169	174	279	284	289	294	299	404	409	414	419	424
30	35	40	45	50	155	160	165	170	175	280	285	290	295	300	405	410	415	420	425
51	56	61	66	71	176	181	186	191	196	301	306	311	316	321	426	431	436	441	446
52	57	62	67	72	177	182	187	192	197	302	307	312	317	322	427	432	437	442	447
53	58	63	68	73	178	183	188	193	198	303	308	313	318	323	428	433	438	443	448
54	59	64	69	74	179	184	189	194	199	304	309	314	319	324	429	434	439	444	449
55	60	65	70	75	180	185	190	195	200	305	310	315	320	325	430	435	440	445	450
76	81	86	91	96	201	206	211	216	221	326	331	336	341	346	451	456	461	466	471
77	82	87	92	97	202	207	212	217	222	327	332	337	342	347	452	457	462	467	472
78	83	88	93	98	203	208	213	218	223	328	333	338	343	348	453	458	463	468	473
79	84	89	94	99	204	209	214	219	224	329	334	339	344	349	454	459	464	469	474
80	85	90	95	100	205	210	215	220	225	330	335	340	345	350	455	460	465	470	475
101	106	111	116	121	226	231	236	241	246	351	356	361	366	371	476	481	486	491	496
102	107	112	117	122	227	232	237	242	247	352	357	362	367	372	477	482	487	492	497
103	108	113	118	123	228	233	238	243	248	353	358	363	368	373	478	483	488	493	498
104	109	114	119	124	229	234	239	244	249	354	359	364	369	374	479	484	489	494	499
105	110	115	120	125	230	235	240	245	250	355	360	365	370	375	480	485	490	495	500

Name ______________________________

Address ______________________________

City ____________________ State ________ Zip ________

Kilobaud MICROCOMPUTING • January 1981

Microcomputing • POB 2741 • Clinton IA 52735

BOOKS

Please send me the following Microcomputing products:

Qty.	Catalog #	Title	Unit Price	Total

Add $1 shipping/handling ________

Total ________

Please allow 4–6 weeks for delivery. No C.O.D.s accepted.

Enclosed $ __________ ☐ Check ☐ M.O.
Bill: ☐ AE ☐ MC ☐ Visa

Card # ____________________ Exp. date ________
Signature ________________ Interbank # ________
Name ______________________________
Address ______________________________
City ______________________________
State ____________________ Zip ________

1/81

Microcomputing • Mail Order Dept. • Peterborough NH 03458

SUBSCRIPTION

MICROCOMPUTING subscribers save $10 off the newsstand price.

☐ New subscription ☐ Renewal
☐ 1 year —$25
☐ 2 years—$38
☐ 3 years—$53

Enclosed $ __________ ☐ Check ☐ M.O.
Bill: ☐ MC ☐ Visa ☐ AE ☐ me

Card # ____________________ Exp. date ________
Signature ________________ Interbank # ________
Name ______________________________
Address ______________________________
City ____________ State ____________ Zip ________

Canada—$27. 1 year only, US funds
Other foreign—$35, 1 year only, US funds

311B9

Microcomputing • POB 997 • Farmingdale NY 11737

kilobaud MICROCOMPUTING

NO POSTAGE NECESSARY IF MAILED IN THE UNITED STATES

BUSINESS REPLY CARD

FIRST CLASS PERMIT NO. 217 CLINTON IA 52735

POSTAGE WILL BE PAID BY ADDRESSEE

kilobaud MICROCOMPUTING T.M.

POB 2741
Clinton IA 52735

NO POSTAGE NECESSARY IF MAILED IN THE UNITED STATES

BUSINESS REPLY CARD

FIRST CLASS PERMIT NO 1024 PETERBOROUGH NH 03458

POSTAGE WILL BE PAID BY ADDRESSEE

kilobaud MICROCOMPUTING T.M.

Subscription Dept.
POB 997
Farmingdale NY 11737

NO POSTAGE NECESSARY IF MAILED IN THE UNITED STATES

BUSINESS REPLY CARD

FIRST CLASS PERMIT NO 1024 PETERBOROUGH NH 03458

POSTAGE WILL BE PAID BY ADDRESSEE

kilobaud MICROCOMPUTING T.M.

Peterborough NH 03458

Att. Mail Order

LETTERS

(from page 24)

I am puzzled why any surplus—positive or negative—is not simply returned (deducted from) to the principal and thus earn income.

The hypothesized 30-year duration of the home mortgage is not used. In fact, it is not needed, since the duration is completely determined by the principal, interest rate and payment. However, the program does not use this, and we find the curious phenomenon where the equity exceeds the value. (In actual fact, the widow's cost of living is reduced—often an important item in estate planning.) Correction of this item would require some programming changes around line 310.

Albert Nijenhuis
Philadelphia, PA

Any economic model like this one must make explicit and implied assumptions that must also be changed often to fit actual new situations and/or the need for revised projections (especially in complex projections such as estate planning or the American national budgeting process). But, Mr. Nijenhuis' suggestions are interesting and will be given serious consideration in my continuing revision of the estate program.

Dr. James Owens
Washington, D.C.

Heath's Sales

In the October "Publisher's Remarks" (p. 6) you state that Heath lost "tens of millions of dollars in sales" by not using the S-100 bus. Actually, Heath has virtually cornered the market for the kind of microcomputer they sell—the mainframe build-it-yourself computer. No one else can match their price for a complete mainframe, power supply, CPU and intelligent front panel (under $300 for the H8). Their bus is free of the glitches and incompatibility problems that have plagued the S-100, and there are plenty of boards for it from various manufacturers.

As for lack of advertising, anyone who is inclined to build electronic kits knows about Heath. They got off to a slow start in software, but they now have a good selection of that as well. Their sales may be small compared to Radio Shack's, but they are not really competing with them.

I would also like to correct a couple of bugs in my article "Upgrading the Heath H8 with a Z-80" (October *Microcomputing*, p. 50). In the caption for Fig. 1, the word "isn't" should be changed to "is." Also, I failed to mention that even if the pins are shortened on the adapter, the CPU board will still not fit in the first slot of the motherboard unless the speaker wire connectors on the back of the front panel board are shortened or removed.

After I wrote the article, I designed a PC board version of the adapter and was planning to build a few and sell them, but I have had trouble making the boards. I would welcome suggestions (or questions about my article) from readers.

Patrick Swayne
2155 Welch Dr.
Stevensville, MI

Refreshing

I read with great interest Wayne Green's "Publisher's Remarks" in the October issue. This was the first opportunity I have had to look at your magazine, and it was a refreshing introduction to read Wayne's article. I thought his articles "Five Years Ago, Five Years Hence" and "The Detroit Syndrome" are classics.

I have been in the process of attempting to select a computer for my law office to do the things I want it to do today and yet have the expandability of taking care of next year and the following year's work. I have written to several of today's "leading manufacturers" indicating my interest and needs and uses for which we want to put the computer but have received no response from the "big ones." It is refreshing to note, however, that the smaller companies in the field are more than happy to respond.

Please continue with these articles, and I am looking forward to seeing the forecasted effect the Japanese market will have on the American firms. It is not that we need such a market, but competition seems to do a lot of good things to the big giants, as has been demonstrated in the automobile business today. Thank you for writing an article which I can understand and particularly for addressing the ordinary reader rather than the genius who normally works with computers.

Malcolm H. Aukerman
Newport, IN

TRS-80 to the Rescue

We are a young and small manufacturer of electronic controls, competing with many large corporations for business in the marketplace. We have invested in a Radio Shack TRS-80 Model I, with 48K, disk drives and printer. We have had this microcomputer on line for about one year. Since that time, we have placed our bookkeeping on the computer to age payables/receivables without hourly wage costs and report formats. This has saved us approximately $2500 per year in outside bookkeeping costs.

Additional services include simple word processing to provide technical manuals, installation/operation instructions and list pricings. Since these items require managerial/engineering time to change, considerable money has been saved simply through the ability to make small changes without starting all over again.

As a small corporation in the high growth stage of its life, we are constantly updating sales projections, cash flow projections, etc. This required literally days of managerial time in revision and specification. With the Visicalc program (recently introduced by Radio Shack), I was able to project cash flow in a matter of hours. This has further enhanced the computer's ability to pay for itself.

This is not to say that the microcomputer is a panacea for all corporations. But the computer has allowed us to remain at lower overhead than our competition and continue the "small company" ability to respond quickly. The low price tag for the system (approximately $3500, including software) has allowed us to afford a system which we could not otherwise.

Without the computer, we would still be in business and still be performing many of the same functions we are now, but at higher overhead cost. The computer has allowed us to reduce that increase in overhead.

Marc Hanlan
President
MM&T, Inc.
Appleton, WI

Software Dilemma

Your editorial in the November issue ("Business Microcomputers: Still a Rip-off?" p. 6) shows unusual insight concerning the microcomputer software dilemma. Most off-the-shelf software is so defective, we refuse to even evaluate new offerings. It costs us an average of five times the retail price to thoroughly evaluate a program. We will never buy software without code, not because we wish to steal it, but because we end up debugging it at our cost.

The small-business person has not been dealt a full deck, and computer firms with high integrity cannot afford to help him. I cannot even afford the luxury of explaining the difference between a sophisticated and a low-cost micro. The business person looking at the low-cost unit will want to hold me accountable for his software problems, and life is too short. Five hours spent selling a $30,000 installation to a potentially happy customer is far more gratifying than spending 50 minutes selling $5000 of disenchantment.

James E. Knowles
President
Synergistics International Ltd.
Elgin, IL

Edited by Dennis Brisson

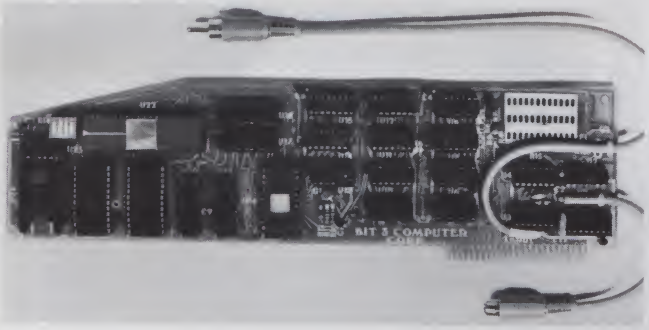
Motorola's 8-bit microprocessor
6502 Microcomputer
Dot matrix printer
Real-time clock for Apple and TRS-80

80-Column Apple II Card

The Full-View 80 is an 80 × 24 upper/lowercase plug-in card that provides 80-column capability for the Apple II computer while retaining the standard Apple II 40-character and graphics operating mode. You can select the different displays via a keystroke or under program control. A 7 × 9 dot character size model and a 5 × 7 dot character size model, which is suited for 8 MHz and other low-performance monitors, are available. Both have upper and lowercase characters. Custom-designed character sets are available via two EROM-type versions (127 and 255 characters).

On-board 2K ROM provides full keyboard editing, complete cursor control and tabbing. Several operating modes are available. The firmware incorporates Pascal and BASIC protocols so you don't have to enter machine-language programs or change Pascals, miscellaneous information or GOTOXY files. Price is $395.

Bit 3 Computer Corporation, 1890 Huron St., St. Paul, MN 55113. Reader Service number 477.



80 × 24 Apple II display card from Bit 3 Computer Corp.

Dot Matrix Printer

The DIP-85 Data Impact Printer features 7 × 7 or 14 × 7 dot matrix, six different character sizes, 100 cps bidirectional print speed, selectable tractor or friction paper feed and long-life ribbon cartridge loading. It has variable line density and continuous form length controls. It can be operated at 100 percent duty cycle continuously without overheating.

DIP-85's high-resolution dot addressable graphic capability can provide plotting, CRT screen graphics, illustrations and special-effect symbols. Its telecommunication capability includes standard baud rate up to 9600, parallel, serial RS-232C as well as X-on X-off transmission control and a standard 1K buffer.

With a full 96-character ASCII set, it is capable of both uppercase and lowercase printing at 80, 96 or 132 characters per line on 8 1/2-inch wide paper. Paper feed is at the rate of 10 lines per second. Operator control includes power, select/deselect, line feed, top of form, self test and variable vertical tab setting. The 17 × 9.75 × 6.5 inch printer costs $895.

DIP, Inc., 745 Atlantic Ave., Boston, MA 02111. Reader Service number 478.

The DIP-85 dot matrix printer.

CMOS Processor

The MC146805E2 is a new eight-bit CMOS microprocessor—the first in a series of planned CMOS microcomputer parts announced by the MOS Integrated Circuits Division of Motorola, Inc., 3501 Ed Bluestein Blvd., Austin, TX 78721. It has 61 basic instructions that are similar to those of the popular MC6800 microprocessor, plus a complete set of bit-manipulation instructions to allow any bit in RAM or any I/O pin to be individually set or cleared with a single instruction.

The low power requirement, 20 mW at 1 MHz and less than 1 mW in standby, is useful for those applications where power is a major consideration (portable instruments, telecommunications, point-of-sale terminals, appliance controllers). The voltage range is three to six volts.

On-chip functions include an eight-bit timer with software programmable seven-bit prescaler, 112 bytes of RAM and a clock generator. The multiplexed bus has an 8K

The Motorola MC146805E2 eight-bit microprocessor.

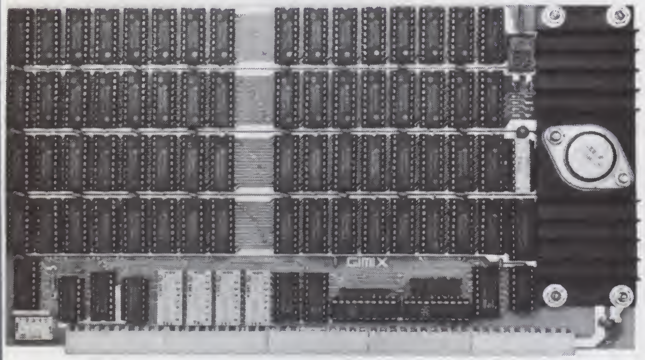
LOOK

WHAT'S COOKING on the FIFTY BUS

32K STATIC RAM BOARDS

Designed for use with:

★ **Existing SS50 Systems** ★ **SS50C Extended Address Systems**



- Assembled
- Burned In
- Tested

16K . . . $328.12

24K . . . $438.14

32K . . . $548.15

16K and 24K Versions are socketed for 32K and require only additional 2114's for expansion.

FEATURES:

- Decoding for 4 Extended Address Lines (allows memory decoding up to 1 megabyte)
- DIP-switch to set extended addressing or disable it
- 4 separate 8K blocks, addressable to any 8K boundary by DIP-switch
- Each 8K block may be individually disabled
- Write protect either of two 16K sections
- Low power consumption — uses 2114L low power RAMS
- Fully Socketed
- Gold Bus Connectors
- Guaranteed 2MHz operation

AND NOW . . . GIMIX OFFERS YOU A Choice of 6800 or 6809 CPU CARDS

You can order your system to fit your needs or select one of the below featured systems. Please contact the factory for further information and availability.

Add as much memory as you need using **GIMIX Static RAM Cards** for the utmost in reliability.

32K 6800 SYSTEM $1,694.59
Includes: Chassis, 6800 CPU, 32K RAM BOARD, I/O card

32K 6809 SYSTEM $1,844.69
Includes: Chassis, 6809 CPU, 32K RAM BOARD, I/O card

32K 6809 PLUS SYSTEM $1,994.79
Includes: Chassis, 32K RAM BOARD, I/O Card, and features our 6809 PLUS CPU Card with the Time of Day Clock option with battery back-up installed, as well as the 6840 Timer Package that provides 3 independent 16 bit counters.

This system also allows the following options to be added at additional cost:

- Battery back-up of the 1K RAM by substituting CMOS parts.
- A 9511 or 9512 Arithmetic Processor.
- GIMIX or SWTP Dynamic Address Translators.

EXPORT NOTES:

For 50Hz 230V C.V. POWER SUPPLY . . . . Add $30.00
80 x 24 VIDEO BOARDS — Specify Format (No Added Charge)

On Orders under $250.00 for a Single Board, or Chips, please Add $30.00 Handling and we will ship Air Mail Prepaid. On all other orders we will ship via Emery Air Freight Collect, and we will charge no handling. All orders must be prepaid in U.S. Funds. Please note that foreign checks have been taking about eight weeks for collection, so we would advise wiring money or checks drawn on a bank account in the U.S. Our bank is the Continental Illinois National Bank of Chicago, Account #73-32033. Visa or Master Charge also accepted.

FACTORY PRIME STATIC RAMS

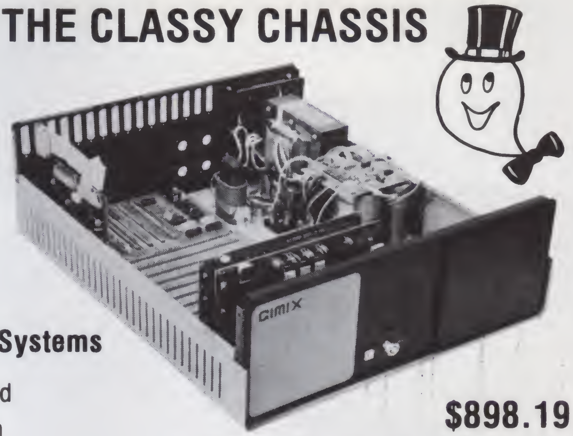
2114L 450 ns . . $5.90 300 ns . . $6.40 200 ns . . $6.90

4044 450 ns . . $5.90 250 ns . . $6.90

ADD $5.00 HANDLING ON ORDERS UNDER $200.00

GIMIX® and **GHOST**® are Registered Trademarks of GIMIX INC.

THE CLASSY CHASSIS



$898.19

- 25 amp (5V) ferro-resonant constant voltage power supply,
- Heavy weight aluminum cabinet with 3 position key switch, fan, and provisions for two 5'' disk drives;
- 6800/6809 Mother Board, fifteen 50 pin and eight DIP-switch addressable 30 pin slots (gold plated pins), fully decoded;
- Baud rate generator on I/O section of Mother Board.

I/O BOARDS

for the 30 PIN BUS:

1 Port Serial . . . . (RS 232 or 20MA, current loop)	$ 88.41
2 Port RS 232 Serial	128.43
2 Port Parallel	88.42

for the 50 PIN BUS:

8 Port RS 232 Serial	288.40
8 Port RS 232 Serial . . . . with on board Baud Rate generator.	318.46
8 Port Parallel	198.45

BOTH 6809 SYSTEMS FEATURE OUR NEW TERMINAL BASED GMXBUG 09 SYSTEM MONITOR

GMXBUG 09 includes advanced debugging tools, utility, and memory manipulation routines.

Both 6809 Systems:

★ Can be reconfigured to allow use of other system monitors (OS-9 and SBUG-E)

★ Include 1K of Scratchpad RAM on the CPU

★ Allow optional software switching of system monitcrs.

2MHz 6809's at slight additional cost when they become available.

Phone, write, or see your dealer for details and prices on our broad range of Boards and Systems for the SS50/SS50C bus and our AC Power Control Products for all computers.

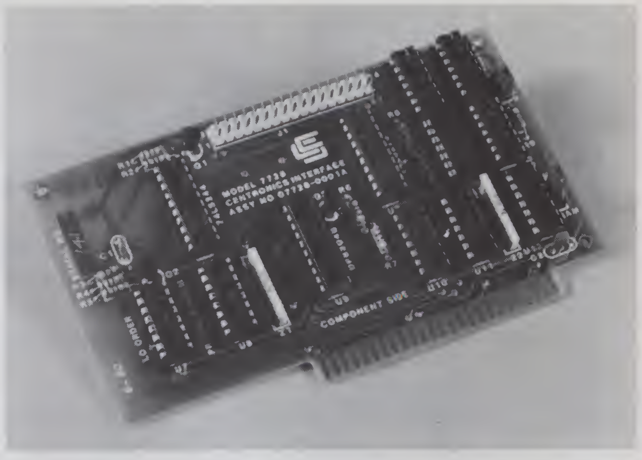
GIMIX INC.

✓22

The Company that delivers
Quality Electronic products since 1975.

1337 WEST 37th PLACE, CHICAGO, IL 60609
(312) 927-5510 • TWX 910-221-4055

✓ *Reader Service—see page 226*



The Model 7728 from California Computer Systems.

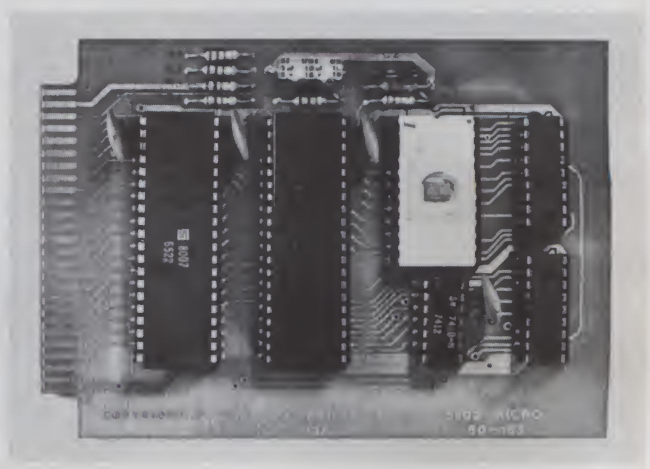
byte addressing range. A 2K byte CMOS ROM, the MCM65516, is a companion part also available. Price is $45. Reader Service number 480.

6502 Microcomputer

The 6502 Microcomputer from John Bell Engineering, PO Box 338, Redwood City, CA 94064, features 1024 bytes of RAM (two 2114s), 2048 bytes of EPROM (2716) and uses one 6522 VIA, which provides two 16-bit programmable timer/counters, serial data port and latched output and input with handshaking logic. It is TTL and CMOS compatible. It is designed to control functions such as temperature, burglar alarms and lights, as well as home appliances, and is also being used for controlling solar heating and power systems and irrigation systems. Prices are $110.95 for the assembled unit, $89.95 for the kit and $24.95 for the bare board. Reader Service number 484.

Apple II Printer Interface

The Model 7728 Centronics printer interface gives Apple II users compatibility with printers using Centronics-type parallel interfaces, such as the Integral Data Paper Tiger, the Okidata Microline 80, the Microtek MT-80P and the MPI 88T, as well as Centronics printers. An on-board 256-byte ROM provides driver firmware and controls ASCII character output to the printer. The driver responds to standard Apple II printer commands for easy selection of command characters, characters per line, auto feed and video echo. Users who choose to develop their own drivers may replace the standard



John Bell Engineering's 6502 Microcomputer.

The Microspeed Language System from Applied Analytics.

ROMs and RAMs. A ROM/RAM jumper makes the necessary logic changes.

The 7728 may reside in any Apple II peripheral slot 1-7. It supports the interrupt daisy chain with arbitration logic including jumper-selectable IRQ generation, and provides DMA daisy chain pass-through. The printer interface includes an eight-bit data output bus, four status inputs, data strobe and acknowledge handshake signals and a printer reset signal. Price is $119.95; cables are available separately.

California Computer Systems, 250 Caribbean Drive, Sunnyvale, CA 94086. Reader Service number 481.

Microspeed System for Apple II

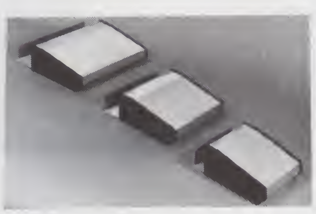
The Microspeed Language System is a hardware/software package that enables Apple II users to run programs at six to 60 times faster than Applesoft BASIC. It employs the Am9511 Arithmetic Processor in conjunction with an interactive compiler to provide increased performance for the Apple II and II+.

The disk-based system offers an enhanced set of programming capabilities beyond those of Applesoft, including print formatting, fast high-resolution plotting, "turtle" graphics and extended, high-speed mathematical functions. The system preserves the desirable features of the Forth concept, including extensibility, inherent program structuring, ease of software development and compactness of compiled code. System utilities include a text editor and mini-assembler; either or both may be loaded optionally from disk. The $495 price includes the auxiliary processor card, Applesoft replacement ROM, 120-page user manual and system software diskettes.

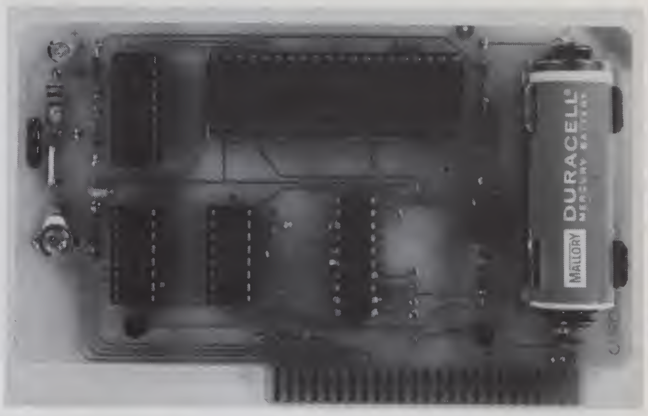
Applied Analytics, Inc., 5406 Roblee Drive, Suite 500, Upper Marlboro, MD 20870. Reader Service number 476.

Computer Enclosures

A new designer series of desk-top enclosures to accommodate computer equipment has been introduced by Jameco Electronics, 1355 Shoreway Road, Belmont, CA 94002. Their four-piece construction enables easy access for servicing while providing strong protection. The end pieces are precision-mold high-strength epoxy with an internal slot (all around) to accept both top and bottom panels, which are aluminum. The enclosures are available in three width sizes: eight inches ($29.95), 10.65 inches ($32.95) and 14 inches ($34.95). The height is 3.15 inches, and the depth is 8.25



Jameco Electronics' computer enclosures.



Frisbee Electronic's Apple II clock module.

inches. Reader Service number 483.

Real-Time Clock for Apple II

The Chrontronix ACV-1 is a real-time clock for the Apple II that includes complete clock features (hr., min., sec., day, mo., yr.) plus programmable interrupts. Slot independent firmware which allows easy access from BASIC (three program lines) is provided. A set of software programs is also available on either cassette or disk to execute dating of forms, stopwatch functions and time conversions.

Standard features include power-down ROM, high-accuracy crystal and installed battery. Sockets are provided for either the standard firmware or 256 bytes of user ROM. Price is $109.

Frisbee Electronics, PO Box 556, Ridgecrest, CA 93555. Reader Service number 474.

TRS-80 Real-Time Clock

Now your Level II TRS-80 can perform real-time operations without the expansion interface with the RTC-10 from Taylor Engineering, 7805 Michael Lane, Huntsville, AL 35802. The RTC-10 runs continuously, keeping accurate time without interfering with cassette functions. You may program events for the same period with real-time interrupts (one per second), which permits preprogrammed activities to take place without derailing on-going programs.

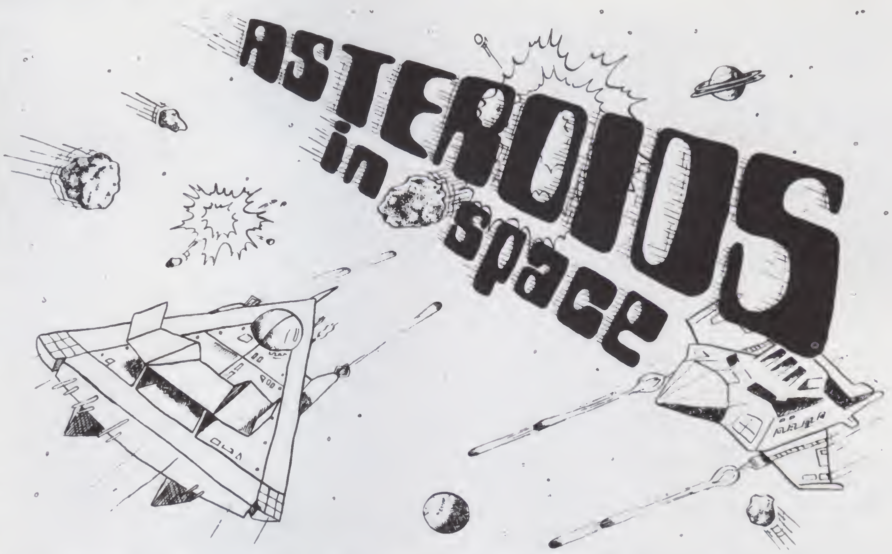
It features 12- and 24-hour operating modes, 12-hour mode AM/PM indicator bit and continuous display mode (continuously displaying time in the upper right hand corner of the screen). All modes are software controlled. The RTC-10 mounts securely inside the TRS-80 keyboard unit and is easy to install and remove without altering the original design of your computer. Price is $25. Reader Service number 475.

Keyboard Enhancer

With the Keyboard Enchancer from Videx, 897 N. W. Grant Ave., Corvallis, OR 97330, you can directly enter upper and lowercase characters from your Apple II keyboard. In addition to the original keyboard entry mode, the typewriter mode and the shift lock mode are keyboard selectable. In both of the last modes, the shift keys will perform exactly as they do on any typewriter.

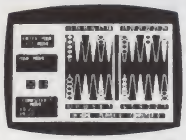
The Keyboard Enhancer allows you to enter nine new characters directly from your keyboard utilizing the shift keys in conjunction with other alphabetic keys. A new power key cap is included with two built-in LEDs for instant positive identification of which mode you are in. The control key must be depressed with the reset key to activate the reset operation. The kit includes five ICs mounted on a PC board, the necessary mounting screws, a jumper cable, power key cap with LEDs and cable assembly and instructions. Price is $87. Reader Service number 482.

Software for the Apple II and Apple II Plus*



ASTEROIDS IN SPACE™
By Bruce Wallace
An exciting space action game! Your space ship is traveling in the middle of a shower of asteroids. Blast the asteroids with lasers, but beware — big asteroids fragment into small asteroids! The Apple game paddles allow you to rotate your space ship, fire its laser gun, and give it thrust to propel it through endless space. From time to time you will encounter an alien space ship whose mission is to destroy you, so you'd better destroy it first! High resolution graphics and sound effects add to the arcade like excitement that this program generates. Runs on any Apple II with at least 32K of RAM and one disk drive.

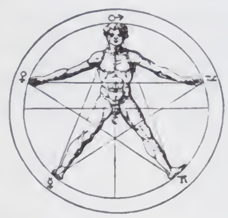
On Diskette - $19.95



FASTGAMMON™ By Bob Christiansen.
Sound, hi res, color, and musical cartoons have helped make this the most popular backgammon playing game for the Apple II. But don't let these entertaining features fool you — FASTGAMMON plays serious backgammon. Runs on any Apple II with at least 24K of RAM.

Cassette - $19.95 Diskette - $24.95

ASTROAPPLE™ by Bob Male.
Your Apple computer becomes your astrologer, generating horoscopes and forecasts based on the computed positions of the heavenly bodies. This program offers a delightful and stimulating way to entertain friends. ASTROAPPLE produces natal horoscopes (birth charts) for each person based on his or her birth data. Any two people may be compared for physical, emotional, and intellectual compatibility. The program is written in Applesoft BASIC with machine language subroutines. It requires either RAM or ROM Applesoft and at least 32K of memory.

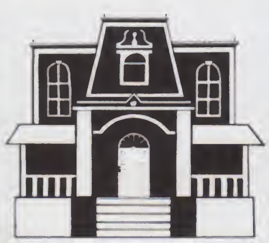


Cassette - $14.95 Diskette - $19.95

FRACAS™ by Stuart Smith.
A fantastic adventure game like no other! Up to eight players can participate in FRACAS at the same time. Journey in the land of FAROPH, searching for hidden treasure while warding off all sorts of unfriendly and dangerous creatures. You and your friends can compete with each other or you can join forces and gang up on the monsters. Your location is presented graphically and sound effects enliven the battles. Save your adventure on diskette or cassette and continue it at some other time. Both integer BASIC and Applesoft versions included. Requires at least 32K of RAM.

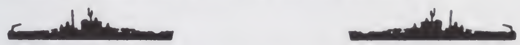
Cassette - $19.95 Diskette - $24.95

BENEATH APPLE MANOR™ by Don Worth.
Descend beneath Apple Manor into an underground maze of corridors, rooms, and secret passages in quest of rich and powerful treasures. The dungeon complex consists of many floors, each lower level more dangerous than the last. If you can reach the lowest level, you may even find the ultimate object of your quest, the fabled golden apple of Apple Manor. Strategy is extremely important as you deal with a variety of monsters, each with its own characteristics. Written in integer BASIC with machine language subroutines. Requires integer BASIC and at least 16K of RAM on cassette or 32K of RAM on diskette.



Cassette - $14.95 Diskette - $19.95

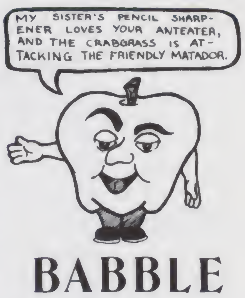
BATTLESHIP COMMANDER™ by Erik Kilk and Matthew Jew.



A game of strategy. You and the computer each start out by positioning five ships of different sizes on a ten by ten grid. Then the shooting starts. Place your volleys skillfully — a combination of logic and luck are required to beat the computer. Cartoons show the ships sinking and announce the winner. Sound effects and flashing lights also add to the enjoyment of the game. Both Applesoft and integer BASIC versions are included. Requires at least 32K of RAM.

Cassette - $14.95 Diskette - $19.95

BABBLE™ by Don Worth.
Have fun with this unique software. You write a story, entering it as a BABBLE program. As you write the story you specify certain words to be selected by the computer or entered from the keyboard at execution time. Run the program and watch BABBLE convert your story into an often hilarious collection of incongruities. The ways in which BABBLE can entertain you are limited only to your imagination. You can compose an impressive political speech or write poetry. You can plan a dinner menu. You can even form images on the screen or compose musical tunes with the help of BABBLE. The cassette version requires at least 16K of RAM and the diskette version requires at least 32K of RAM. BABBLE is written in machine language and runs on any Apple II computer.



Cassette - $19.95 Diskette - $24.95

L I N K E R

LINKER by Don Worth.
Turn your Apple II or Apple II Plus into a powerful and productive software development machine with this superb linking loader/editor package. LINKER does the following and much more:

- Dynamically loads and relocates suitably prepared machine language programs anywhere in RAM.
- Combines a main program with subroutines. You can assemble a subroutine once and then use it with as many main programs as you wish.
- Produces a map of all loaded routines, giving their location and the total length of the resulting module.
- Contains a library of subroutines including binary multiplication and division, print text strings, delay, tone generator, and random number generator.

Linker works with virtually any assembler for the Apple II. Requires 32K of RAM and one disk drive.

Diskette - $49.95
Manual Only - $19.95

QS QUALITY SOFTWARE
6660 Reseda Blvd., Suite 105, Reseda, CA 91335
(213) 344-6599

Now exclusive distributors for products from The Software Factory, Newhall, California
*Apple II and Apple II Plus are trademarks of Apple Computer, Inc.

WHERE TO GET IT: Call us at (213) 344-6599 for the name of the Quality Software dealer nearest you. If necessary you may order directly from us. Mastercard and Visa cardholders may place orders by telephone. Or mail your check or bankcard number to Quality Software, 6660 Reseda Blvd., Suite 105, Reseda, CA 91335. California residents add 6% sales tax. SHIPPING CHARGES: Within North America orders must include $1.50 for first class shipping and handling. Outside North America the charge for airmail shipping and handling is $5.00. Pay in U.S. currency.

✓Reader Service—see page 226

Edited by Dennis Brisson

Accounting system, PET/CBM graphics, Spelling dictionary, Bowling program

Accounting System

A complete accounting system for microcomputers with two eight-inch disk drives, standard CP/M, an 80×24 character terminal and Microsoft BASIC 5 is now available from Houston Micro-Computer Technologies, 5313 Bissonnet, Bellaire, TX 77401. The system functions as a single entry system; all transactions are passed automatically to the General Ledger. Departmental and master/sub accounts are easily used. Account numeric ranges, the income statement and balance sheet can be adjusted to match your present system. Check printing is easily done with both Accounts Payable and Payroll. Open invoice, balance forward and auto billing are all possible with Accounts Receivable. Invoices, statements, customer lists and aging reports are also easily produced. Reader Service number 491.

TRS-80 Utility Package

Racet Computes, 702 Palmdale, Orange, CA 92665, has recently introduced a utility package for the TRS-80 Model II system. It provides eight new DOS commands, and features the ability to:

reconstruct or recover data from bad diskettes
copy multiple files
examine/change diskette contents
catalog diskette directories
change disk names and create files

The entire package is written in machine language and is fully documented in a 124-page manual. Price is $150. Reader Service number 492.

Mail List

Mail List lets you easily prepare and organize mailing list labels and sort them four ways: alphabetically, as inactive or active, by zip code and by a designated six-character utility field. Designed for use with the Commodore CBM 16K and 32K computers with CBM 2040 disk drives and CBM or ASCII printers, Mail List stores up to 1050 records on a single disk. You can easily adjust the length of all fields. Price is $95.

CDS Corporation, 695 East Tenth North, Logan, UT 84321. Reader Service number 490.

Apple Terminal Emulator

Stem is a terminal emulator for connecting an Apple II to any mainframe computer at speeds to 1200 bits per second. It includes a switch to print all data as received or entered, converts lowercase to uppercase, discards unused control codes and generates BREAK with a single key input and auto restart on RESET. Support exists for standard Apple video and communications and parallel printer cards. Price is $15, tape or disk.

Video Business Systems, 59 Noyes St., Concord, NH 03301. Reader Service number 496.

6809 Word Processing System

Stylograph is a word processing system for 6809 computer systems operating under the Flex and OS-9 operating systems. The program features cursor-based editing with on screen formatting, which simplifies word processing tasks since there is no need for intermediate printouts to verify appearance of the text. Stylograph has a full array of editing and formatting commands. Prices are $150 for specialty printer versions and $135 for TTY-type printers.

Sonex Systems, Box 238, Williamsville, NY 14221. Reader Service number 487.

Service Analysis Tracker

The Service Analysis Tracker (SAT) is designed to analyze the operation of service-type operations (retail stores, counter service, vending) and determine the optimum scheduling strategy to maximize usage of people and/or machines. It uses a sampling system to determine times of the day when employees and/or equipment are actually needed for customer service. SAT determines when it is economical to staff and/or equip your operation in light of customer activity. It features a graphic overlay capability so you can see week-to-week and day-by-day trends. The program is available for the 48K Apple II ($219) and for 48K CP/M systems with addressable cursor terminals ($325).

Stoneware Microcomputer Products, 1930 Fourth Ave., San Rafael, CA 94901. Reader Service number 494.

PET/CBM Graphics Package

Curve is a graphics software package for 32K PET/CBM systems. It features capabilities to plot parametric, Cartesian or polar equations; data points or characters entered from the keyboard; shaded bar graphs in the vertical or horizontal orientation; linear and logarithmically scaled axes; and fully scalable alphanumeric character sets. Sixteen subroutines are included for use in custom programs. Curve incorporates built-in error messages that trap incorrect input quantities. It is designed to be used with the new Watanabe Digiplot plotter via the Com-plications, Inc., C102, IEEE-488 to Watanabe interface.

ECX Co., 2678 North Main St. #6, Walnut Creek, CA 94596. Reader Service number 488.

Spelling Dictionary for CP/M Users

WordSearch is an automated spelling word dictionary system that can help you eliminate misspelled words from your documentation, letters and other correspondence. Designed for use with MicroPro's Word Star and other users of CP/M-compatible word processing systems,

WordSearch quickly processes any text file, isolates a list of the unique words used, compares these words to a library of correctly spelled words and identifies any words used that were not found in the spelling dictionary. It then allows any valid words encountered to be automatically added to the spelling dictionary. WordSearch also locates the words within your text and clearly identifies them for rapid correction. Available on eight-inch single-density diskette. WordSearch costs $195.

Key Bits, Inc., PO Box 592293, Miami, FL 33159. Reader Service number 489.

OSI Word Processor Justification

Now justification is available for the WP6502 word processor from Dwo Quong Fok Lok Sow with the DQ Justify I package. DQ Justify I allows the addition of TTY justification and centering at all ports and micro spacing justification with underlining for the Parallel NEC Spinwriter.

Price is $50 for either the OS65D or OS65U version on eight-inch or five-inch diskettes. It will install itself on the WP6502 master disk.

Dwo Quong Fok Lok Sow, 23 East 20th St., New York, NY 10003. Reader Service number 485.

Heath/Zenith Software

A new line of software tools for word processing, software development and time-sharing access for Heath/Zenith 8080/Z-80 computers has recently been introduced by The Software Toolworks, 14478 Glorietta Drive, Sherman Oaks, CA 91423.

Word processing is provided by the PIE 1.5 full screen text editor and the TEXT text formatter. PIE allows 24 function key commands and cursor motion keys that allow editing anywhere on the screen.

The C/80 C Compiler implements a large subset of the C programming language and includes a run-time I/O library. Also available are Z-80 and 8080 op-code versions of the UVMAC macro assembler, the REACH modem and File Transfer program, the PACK file compressor and CRYPT file encryptor.

This software (priced from $20–$40) runs on the H89, Z89 and H8 (with H19) computers under the HDOS operating system in 32K. Reader Service number 498.

Apple II Bowling League Program

The Bowling Data System is a new Apple II program for managing bowling league data and is designed for leagues of up to 40 teams with up to six bowlers per team. A cumulative record is kept of total pins, games won and lost, total points and high series for each team and for each bowler, as well as high game, handicap and other data.

Weekly scores for each bowler are entered from score sheets provided by the system, which then prints a weekly recap report and the score sheets to be used for the following week, automatically generating team and lane pairings. A season average report also is produced by the system, listing all bowlers in the league in descending order by average. The system will run on an Apple II with 32K of RAM, Applesoft in ROM, one disk drive and an 80-column printer. Disk price is $79.95.

Rainbow Computing, Inc., 9719 Reseda Blvd., Northridge, CA 91324. Reader Service number 486.

Apple School Administration Program

The Assistant Principal is a complete administrative package for high schools and junior high schools. The package provides total control of class rosters, student master records, student schedules, teacher assignments and grade reporting. The system allows the school administrator to design student input documents, enter students onto the system, schedule their classes, print class rosters, accept grades and test scores, print report cards, print file folder labels and prepare student master records. It automatically prints ranked class lists and records attendance information.

The system requires two disk drives and Applesoft in ROM and is provided on seven diskettes for $500.

Monument Computer Service, Village Data Center, PO Box 603, Joshua Tree, CA 92284. Reader Service number 497.

COMPUTER CLINIC

Wanted: OEMs Assembly programmers Stock market and TRS-80 programs

I am preparing for publication a handbook for assembly-language programmers, consisting primarily of quick-reference charts and tables and universal programming tricks. I would appreciate suggestions as to material that assembly programmers would find useful.

Robert Ross
2205 Grayson Place
Falls Church, VA 22043

My company wants to talk to several firms who can assemble those powerful boards advertised in *Kilobaud Microcomputing* into a computer capable of supporting hard disk storage and a number (2–20) of TRS-80 microcomputers in an interactive mode. We would require two to four such systems per month, assuming we can find a knowledgeable OEM to assist us.

G. William Carlson
Automated Medical Systems Inc.
One Buttonwood Square, Suite 205
Philadelphia, PA 19130

As an early retiree who has done well in speculation through the systems approach, I am interested in contacting computerists who are doing advanced forecasting on both stock and commodity markets.

Ted Broder
Box 407
Flushing, NY 11363

As the owner of a 32K (soon to be 48K) TRS-80 Level II and a Pertec TFD-200 77-track disk drive, I am looking for the following programs:
a capital equipment inventory control program
a general inventory program
a word processor for business communication
a preventative maintenance scheduling control program
a database to record maintenance and service information

These programs would have to run under TRS-DOS or on my TRS-80 Level II under Pertec Microdos or under NEWDOS/80.

J. Lehner, c/o SPA
Apt. 191, El Tigre
Edo. Anzoatequi
Venezuela

BOOK REVIEWS

Videotext: The Coming Revolution
Home Computers Can Make You Rich
8085A Cookbook
How to Program Microcomputers

Videotext: The Coming Revolution in Home/Office Information Retrieval
Efrem Sigel, ed.
White Plains, NY: Knowledge Industry Publications, Inc.
Hardcover, 154 pp., $24.95

"In times of chaos, one should always get there first."—Dr. Christine Urban, president of Urban and Associates, at the 125th sales conference of the International Newspaper Advertising Executives last July.

* * * * *

The American communications industry is finally waking up to the fact that electronic information systems are here, and the scene looks like a D. W. Griffith version of the California gold rush. Across the country, corporations of all shapes, sizes and concerns are slavering over the cable stations, teletext systems and computer networks that sparkle in the sun like naked veins of rich, ripe ore. From giant monoliths like Dow Jones and CBS to little operations like the Ottumwa (Iowa) *Courier*, everybody wants to stake a claim.

In view of such volatility and flux, a book on videotext seems like folly. The medium of print is simply too slow to capture the rapid advances being made in the field. Not even the major microcomputing magazines can keep up—the best they can do is keep track.

But remarkably, the authors of *Videotext* have avoided this pitfall. In fact, they have turned a potential weakness into a strength. By concentrating on the history and development of videotext, and by offering an even-tempered discussion of videotext's potentials, Sigel and his friends have managed to step above the fracas and give it some perspective.

The book concentrates mostly on the British systems. This makes sense, for, as Colin McIntyre points out in his article on the CEEFAX teletext service, Britain got a good two years' jump on the rest of the world. The Britons have developed a faculty that comes only with time: hindsight.

The two major systems in Britain (at least, at the time this book was written) are CEEFAX and Prestel, the latter the Post Office's viewdata system. CEEFAX, a service of the British Broadcasting Corporation and therefore publicly owned, went on line in 1976 after a two-year pilot trial. It is a standard, one-way teletext service; that is, a staff of editors and writers keeps a computer full of information on various subjects, while at home the viewer uses a keypad to call it up on his TV screen. Categories include news, finance, sports, food, entertainment, weather and so on.

Prestel, in operation since 1979, is similar to The Source or CompuServe in the U.S. The user retrieves information via the telephone lines, connecting his terminal to the phone with a modem. At present, says Max Wilkenson in his piece on Prestel, subscribers use a simple numerical keypad to call up various items on a menu; he mentions, however, that alphanumeric keyboards and intelligent terminals are in the future.

The authors spend a great deal of time explaining and comparing the potentials, problems and advantages of the two forms of videotext. This is the book's strength; it lays out the facts for its readers without favoring one system over the other. The author's observations can be summed up thusly:

- Teletext systems are far less expensive to set up than viewdata, since they use existing television broadcast signals. CEEFAX, says McIntyre, cost only $200,000 to put on the air, and can be maintained by the BBC for only $200,000 of its annual $300 million budget. This comes to a few pennies per viewer. (Funds are generated by a Broadcast Receiving License each set owner must have.) The Post Office, in contrast, has spent millions on Prestel, and users must pay an hourly charge for the service.

- Prestel offers its subscribers far more information than CEEFAX. "Broadcast teletext should satisfy those who want relatively small quantities of frequently changing information: news, weather, travel, share-prices, sports results," says McIntyre. "For those who want more specialized information for a particular purpose, and want it treated in depth, Prestel is their service."

- To get the most out of viewdata, a user should have, or plan to get, a microcomputer. While micros are rapidly falling within the range of the average person's financial and technical resources, they are far from becoming standard home appliances. Teletext, on the other hand, uses the TV, something nearly every home has at least one of. Telephone availability is a related problem; people who use a viewdata service extensively will need a second phone.

- Both teletext and viewdata must overcome the ingrained attitudes consumers have toward their TV sets. To date, television has been a passive medium, used mostly for entertainment and leisure. Public television has been running up against this problem for years, and there is no reason to believe that videotext will have an easier time of it.

"The real question comes down to consumer need for the service," says Sigel in the concluding chapter. "Obviously there are some individuals addicted to news or financial reports, who absolutely must be plugged into a device that feeds them the latest information. But how many are there, and in what way are they willing to support the service by paying for it?"

Sigel goes on to point out that videotext is not the only new technological wonder available to consumers; it will have to compete with such products as video games and video cassette recorder/players. "Consumers do not want, nor can they afford, every possible gadget," Sigel observes. "They will pick and choose. The choice of a video recorder may mean the rejection of a videotext decoder—not consciously, but because the individual is going to spend his time watching old movies or tennis lessons on the tube, not

Terminals and Printers!

TELEVIDEO TVI-912C

pper and lower case, 15 baud rates: 75 to
9,000 baud, dual intensity, 24 x 80 character
isplay, 12 x 10 resolution. Numeric pad. Pro-
rammable reversible video, auxiliary port,
elf-test mode, protect mode, block mode,
bbing, addressable cursor. Microprocessor
ontrolled, programmable underline, line and
haracter insert/delete. "C" version features
ypewriter-style keyboard. List $950

OUR PRICE: CALL

20C (with 11 function keys, 6 edit keys and
transmission mode keys, List $1030

CALL

Intertec EMULATOR

oftware compatible with a Soroc IQ-120,
azeltine 1500, ADM-3A or DEC VT-52. Fea-
res block mode transmission and printer port;
?" anti-glare screen; 18-key numeric keypad;
ll cursor control. List $895

OUR PRICE $749

NEW INTERTUBE III

List $995 **ONLY $749**

" display, 24 x 80 format, 18-key numeric
eypad, 128 upper/lower case ASCII charac-
rs. Reverse video, blinking, complete cursor
dressing and control. Special user-defined
ntrol function keys, protected and unpro-
cted fields. Line insert/delete and character
sert/delete editing, eleven special line draw-
g symbols.

SOROC

IQ-120

List $995

SPECIAL

$729

IQ-140 List $1495

SPECIAL $1149

HAZELTINE

1500

ONLY

$879

1410 w/numeric keypad, List $900	$749
1420 w/lower case and numeric pad	849
1510, List $1395	1089
1520, List $1650	1389

NEC SPINWRITER™

Terminal/Keyboard as well as
RO Printer Only models available.
CALL FOR PRICES!

CENTRONICS PRINTERS

NEW 730, parallel, friction, tractor	**$649**
NEW 737 parallel, friction, tractor	**$829**
779-2 w/tractor (same as TRS-80 Line Printer I), List $1350	
702 120 cps, bi-direct., tractor, VFU	NEW
703 185 cps, bi-direct., tractor, VFU	LOW
704 RS232 serial version of 703, $2350	PRICES

TI-810

TI-810 Basic Unit, $1895 . **ONLY $1695**

TI-810 w/full ASCII (Lower case), vertical forms control, and compressed print . **$1895**

TI-745 Complete printing terminal with acoustic coupler, List $1695 . . . . **$1399**

PAPER TIGER®

IDS-440 Paper Tiger, List $995	**$695**
w/graphics option, incl. buffer, $1194	**$789**
TRS-80 cable	**45**
NEW IDS PAPERTIGER 460 List $1295	**$1149**
NEW IDS PAPERTIGER 460G List $1394	**$1199**

NEW IDS 460
QUALITY PRINTING AT MATRIX
SPEED—LOGIC SEEKING
PROPORTIONAL SPACING
w/auto text justification

ANADEX
DP9500/DP9501 PRINTERS

DP-9500, List $1650	$1399
DP-9501, List $1650	$1399

OKIDATA Microline 80 ONLY	**$649**
Tractor Feed Option	$99
Serial interface	$89
AXIOM IMP I	**$699**
COMPRINT 912 w/parallel interf.	**$559**
912 w/serial interface, List $699	$589
MICROTEK, List $750	**$675**
ANADEX 80-Col. Dot Matrix	**$849**

Above prices reflect a 2% cash discount (order prepaid prior to shipment). Add 2% to prices for credit card orders, C.O.D.'s, etc. Prices are f.o.b. shipping point. Prices are subject to change and offers subject to withdrawal without notice. **WRITE FOR FREE CATALOG.**

✓50

MiniMicroMart, Inc.

1618 James Street, Syracuse NY 13203 (315) 422-4467 TWX 710-541-0431

VISA · master charge

scanning news bulletins."

Such questions are important these days, at a time when too many people are adopting an unabashed "gee whiz" attitude and painting scenarios for the future that look like something out of the Jetsons. If only Sigel and his colleagues had probed a little further. For instance, what about the actual substance of videotext's offerings—are we simply developing a new form for the same content? Videotext offers more information at greater speeds, but is this what modern man needs when he can't even efficiently process the information he's swamped with now?

And how will videotext affect the social and economic classes of our society? The ghetto-bound, the rural poor and other minorities are not likely to be waiting in line to buy their modems, microcomputers and the like. If knowledge is power, videotext could well increase the gaps between the rich and the poor, the informed and the uneducated.

But this book is probably not the place for such weighty questions. It is enough that *Videotext* provides such an intelligent and self-contained overview of the subject; conjectures on the social ramifications of electronic communications would only have diverted the authors from their intents.

In addition to its sections on CEEFAX and Prestel, *Videotext* includes chapters on videotext in the U.S. and in the rest of the world (the rest of the world in this case comprising Canada, France, Japan, W. Germany, Holland and Scandinavia). While they help to provide a context for developments in Britain, these portions of the book are comparatively outdated and incomplete. The chapter on the U.S., for example, gives only a cursory survey of such systems as Warner's QUBE, Knight-Ridder's Florida experiment and The Source. It makes no mention of CompuServe, and was published too early to catch such events as this country's first electronic newspaper (The Columbus [Ohio] *Dispatch*).

But these inadequacies are not the fault of the authors. It will be a few years before anyone produces a definitive study of videotext. The field is too vibrant to be yet enclosed in a block of plastic, like a ball bearing, and made into a paperweight. In the meantime, *Videotext* serves as a solid introduction, one that will have value for some time to come.

Eric Maloney
***Microcomputing* staff**

Home Computers Can Make You Rich
Joe Weisbecker
Hayden Book Company, Inc.
Rochelle Park, NJ, 1980
Softbound, 128 pp., $5.95

I'm very cynical about get-rich-quick schemes, so I'm usually negative about books such as this one. How many people actually strike it rich after reading how it should be done? I decided long ago that, unless I was incredibly lucky, the only way I was going to get rich—or even earn a comfortable living—was through hard work, and lots of it.

Neither computers nor Joe Weisbecker's book will make you rich. But you have to have something to work with, and Weisbecker has come up with more good ideas than any other author that I've read yet.

This is a book of ideas—an enormous number of ideas. Sell software; sell hardware; do consulting; write articles or books; provide computer services; invent gadgets, interfaces or games—the list goes on and on. He discusses both spare-time and full-time activities. You get the feeling that Weisbecker has successfully tried many of these ideas himself.

At first I felt that he was just skimming along with too many generalizations and not enough specifics. But after awhile I began to realize that there wasn't an idea here that would be used in its entirety or exactly as presented anyway.

Weisbecker gives quite a bit of background on microcomputer hardware and software, dwelling mainly on the inexpensive models such as the KIM, VIP, PET and TRS-80. A person using a computer for serious business purposes is going to want something a little more versatile and sophisticated, but these simple machines are probably a good place for the novice to start.

It rankles me a bit when someone says, as Weisbecker does, that the S-100 bus is obsolete. I contend that it will have a long, vigorous life until someone comes up with something better—and no one has yet. Each new machine on the market, with its new bus that is totally incompatible with all of the previous microcomputers, only increases the cost and complexity of hardware and software. I'll stick with the S-100.

Some of the ideas presented are pretty far-out, but still worth considering. A computer clown for kids' parties? Computer-generated art objects? Show business schemes? And that's only the beginning.

One chapter deals with putting your money to work and is worth reading two or three times. It applies to any money, not just that earned in a computer-oriented business. It doesn't take much mathematical ability to figure out that money in a savings account is shrinking instead of growing because of inflation. But what to do about it? Investing is the answer, and Joe has some suggestions that might help you increase your income, or at least get your thinking pointed in the right direction. Stock market management is one such concept, but there are many more.

When I finish a book, I always ask myself the following question: "Given the price of the book and the material that it contains, can I recommend it as a good buy?" If any one of the hundreds of ideas in *Home Computers Can Make You Rich* earn you as little as $60, you'll have the cost of the book back ten times over.

Should you buy it? If it is your intention to use your microcomputer for anything other than playing games for your own amusement, the answer is "Yes!" If you don't now own a computer, this book might just convince you to rush out and get one.

Rod Hallen
Washington, DC

8085A Cookbook
Titus, Larsen, Titus
Howard W. Sams & Co., Inc.
Indianapolis, IN, 1980
Softcover, 350 pp., $12.95

Designing and building your own microcomputer can be a rewarding experience. But it can also be a frustrating one, particularly when you don't have the slightest idea what you're doing.

The main problem is that, until recently, little has been published on microcomputer design. And what has been printed could only be understood by advanced computer engineers.

This is fortunately changing. The *8085A Cookbook* is one recent attempt to bring microcomputer design down to the hobbyist level.

The authors (known to most as The Blacksburg Group) are well-known for their other books, which discuss various microcomputer topics in a down-to-earth style that is both enriching and easily understood. The *8085A Cookbook* is no exception. Armed with only a basic knowledge of digital electronics (e.g., gates, flip-flops and decoders), the reader can successfully design and build a microcomputer around the 8085A microprocessor.

The book starts off with a discussion of the basic concepts and elements of a microcomputer, such as programming, memory and I/O (input/output). Also included is a detailed description of the 8085A microprocessor, covering its internal architecture, power requirements, control signals, timing and instruction set.

Chapter 2, entitled "Basic System Control," is on the various support circuitry required by the 8085A. It deals individually with such circuits as the clock and control signal decoder, and shows a number of approaches to their design.

The next three chapters concern the design of memory systems. The authors present the basic elements of both static RAM and ROM memory systems, along with the logic circuits needed to make them work. Topics covered include address decoders, read/write signals, access times and data bus buffering. In their designs, the authors use such com-

mon memory ICs as the 2102 and 2114 RAM chips and the 2708 ROM chip, making memory design much easier for the reader.

Chapter 6 contains information on microcomputer interfacing. Here the reader is exposed to I/O operations, I/O ports, USARTs, interrupts and several complex ICs such as the 8255A programmable peripheral interface chip.

Chapter 7 introduces the reader to a number of 8085A family integrated circuits that single-handedly perform tasks that would normally require a number of ICs. This chapter describes the functions provided by each chip and how they are interfaced to the 8085, as well as the advantages and disadvantages associated with each chip.

Chapter 8 ties the previous chapters together in such a way that the reader can actually design and build a completely operational 8085A-based microcomputer. In fact, it includes a complete schematic and parts source for a small trainer computer.

Concluding the book are an 8085A instruction set summary and a profusion of data sheets.

One of this book's qualities is that the authors took the time to explain *why* they chose to use a particular design or IC and the effects of doing it differently. Thus, the *8085A Cookbook* (particularly chapters 3, 4 and 5) should be useful to anyone interested in microcomputer design. I own an 1802-based system and have applied much of what I learned from this book to my computer.

But like most books, the *8085A Cookbook* isn't perfect. For it to truly be a cookbook, one or two chapters should be devoted to software. Instead, the authors refer you to another one of their books, *8085 Software Design* (which, I might add, is well worth having). As it stands now, the reader will be able to build an operational computer, but may or may not have the slightest idea how to program it!

Another improvement would be to include some material on I/O devices—at the very least a hex keypad and an LED readout.

Despite these shortcomings, the book is still well worth having and should prove valuable to any hobbyist who is interested in building or expanding his own computer system, whether it be 8085-based or not.

Steve Dominguez
Golden, CO

How to Program Microcomputers
William Barden, Jr.
Howard W. Sams & Co., Inc.
Indianapolis, IN, 1979
Softcover, 256 pp.

This book deals with the programming of not one, but three of the most popular microprocessors: the 8080, the 6800 and the 6502. It is divided into four sections: Basic Concepts, Microcomputer Architecture and Operation, Assembly Language Programming with Microcomputers and Programming Algorithms. In addition, three appendices list the instruction sets for the three microprocessors.

Basic Concepts includes information on binary and other number systems, arithmetic operations and basic computer operations. The entire section is rudimentary and can be skipped by all but the novice programmer.

Section two tries to cover subjects of more general interest. Six chapters cover microprocessor architecture, addressing modes, memory and stack operations, instruction sets, input/output operations and interrupt processing. The author compares each of the three microprocessors in each treatment of these subjects; when you consider that all six chapters total 38 pages, you get an idea of how lightly each concept is presented.

The third section discusses actual programming and coding techniques. Subjects covered are hand assembly techniques, integer math with multiple precision schemes, subroutines, branching, loops and stack operations. Other chapters cover such concepts as table operations, list processing, decimal and floating point math and input/output drivers. Barden gives examples of each concept for each of the three microprocessors.

The fourth part of the book is simply a collection of subroutines listed in assembly language along with machine codes. Each subroutine is repeated for the 8080, the 6800 and the 6502. They include a double precision shift, a timing routine, single precision multiply and divide routines, multiple precision add and subtract routines, some code conversions (ASCII hex to binary and vice versa, for example), a data fill, a data block move, a string compare, a table search and a random number generator.

This book fills a definite need if you are considering upgrading or switching your present system to one using a different microprocessor. The comparisons made between the three microprocessors and their basic operations can be invaluable in aiding your decision.

It should also prove convenient to anyone who is considering writing a simulator program (for running 8080 programs on a 6800, for example). In fact, in various portions of the book, the author hints how such a program can be implemented.

I don't recommend this book to anyone unfamiliar with machine-language programming; the topics are not covered with enough depth. Therefore, the novice programmer who wants to learn hand assembly techniques will have little success with this book.

Thomas Franks
Wadsworth, OH

PERSPECTIVES

Conclusion

We are using our computers every day in many ways for business, in sales, in research and in manufacturing. Five senior executives have units assigned at home as well as in the office. These have RS-232 boards and acoustic couplers. Besides being usable as business terminals to talk to our big computer at night, they can access MicroNet, The Source and Lockheed's Dialog. And they can be used as word processors or computers to work on those late-night brainstorms.

Several other members of the staff have bought their own TRS-80s for home use. But most staff members trade off and take turns bringing their office TRS-80s home on weekends. Although we have two full-time programmers for our TRS-80s (and some added help from our two programmers for our big computer), nearly one third of all programs are being written by the staff at home on their own time.

So I can only say that the computer bug is catching, even if we businessmen are working on business and scientific programs! ■

PERSPECTIVES

By Henry Lee

I Love My TRS-80s–All 35 of Them

The publisher of *Microcomputing* has thrown down the gauntlet—and someone must respond.

In the March 1980 issue of *Microcomputing Industry*, Wayne Green raised the question of whether microcomputers are really used in business. By the term microcomputers he meant TRS-80s, since over 150,000 TRS-80 Model I's have been sold–more than all other brands put together.

He based this comment on several observations:

- Business software offered by the microcomputer manufacturers is of poor quality.
- Business software sold by independent software houses does not sell nearly as well as games.
- There are very few reviews of business software in the microcomputer magazines.

While I don't necessarily refute Mr. Green's comments, as a businessman-scientist I'd like to cast some light on the subject. I have purchased and supervised the installation of 35 TRS-80 Model I Level II 48K two-disk computers in my company, and I believe I can help answer his questions.

Manufacturers' Software

First, I agree that the software available from manufacturers is not too good. This is true in both the micro and minicomputer business.

We have 11 minicomputers in our company, ranging in size up to 300 megabytes, and 20 terminals. The software support from the manufacturers is no better than the support we get from the microcomputer manufacturers. Most manufacturers of business computers tend to divorce themselves from the software and leave it to the owners to hire their own programming. While some concerns are now offering a type of turnkey service, this is the exception.

The programs that the microcomputer manufacturers do supply are so-so. They are generally not in machine language, and hence run slow. They are not fully thought through.

One business mailing list program, for example, is pretty fair, if you don't mind a seven-hour sort, which can be done overnight. But the print commands print only one label across at a time, and most business mailings are done with labels mounted three across.

Also, the program does not seem to allow for the occasional five-line address, which messes up the register on the labels.

So the user has to end up going into the program and doing extensive reworking. This can be almost as costly as writing the program from scratch, and the user often still has a program inadequate for his particular needs.

Independent Software Houses

The business programs from the independent software houses also suffer from inadequacy. They are written by programmers—very capable programmers—but they usually guess what the businessman wants. Even if they do have a businessman help with the specifications, they are faced with the infinitely varied demands of businessmen for differing reports.

I once thought that we should compromise on programming and use the canned programs. But as I got into computers I found it was a mistake to make your computer function with one hand tied behind its back, which is what you're doing if you don't let these powerful machines give you exactly what you want.

Despite the above shortcomings, I buy about one copy of every independent business program written, just to get ideas for our own programming work. What is $50 or $100 for a commercial program if you expect to spend $1000 to spec and write and debug your own program? A commercial program often gives some rather good suggestions, even if the program, per se, does not meet your business needs.

It is beyond the scope of this article to write about the magnificent utility programs written by some independent software houses. These are intended for the mass hobbyist and business markets, and they are often useful to at least a few people in the company. These programs include NewDOS+ (Apparat), Electric Pencil (Michael Shrayer), Textwriter (Organic Software), Penpatch (Mumford) and Smart Terminal (Softside), to say nothing of Harvard Pennington's book *TRS-80 Disk and Other Mysteries*.

On the other hand, I'm now studying six database management systems for possible use. These newer, user-definable programs may have advantages over self-written programs, and are undoubtedly a big step forward. In this sense, these programs seem as good as ones I have been offered for our bigger computers. But I find that it takes quite a bit of study to learn someone else's program, compared to one for which I have written the specifications and have had a hand in writing and formatting. So I'm not sure that even these newer programs are an answer to the business software problem.

We have had good experience with the Analysis Pad from The Bottom Shelf. On the other hand, we have written three other programs for columnar material because we needed special chemical calculations to be built into the program.

Reviews of Business Software

Why are there no reviews of business software?

First, if the commercially available programs are not very good, why bother to review them? Second, most businessmen are too busy to take the time to write an article. Third, we are writing our own unique programs to meet our own unique needs, so what is there to write about?

(continued on page 241)

Dr. Henry Lee is president and chief scientist of Lee Pharmaceuticals, a manufacturer of polymeric biomedical products. His address is 3543 E. California Blvd., Pasadena, CA 91107.

New muscle for Heathkit Computers

Now with new CP/M® Systems Software for thousands more programs...and new 8-inch disk system for millions more bytes.

The Heathkit All-In-One Computer now has space for 64K of addressable RAM, so you can run bigger, more complex programs.

And our new CP/M® Operating System (Standard ORG-0, Version 2.2) makes thousands of CP/M programs available to you. Heathkit systems can run more, do more, store more than ever before.

A new 8-inch double-sided, double-density disk system, with over 1 million bytes per drive, is now available for Heathkit H-89 and H-8 Computers.

The new 8-inch disk system features soft-sectored disks for IBM® compatibility. It's capable of operating in standard IBM 3740 format. And the 8-inch system can be used in conjunction with 5¼-inch systems. **For compatibility with the rest of the industry, Heathkit Computers may just be the most flexible systems you can buy.**

All Heathkit computers and peripherals are available in money-saving, easy-to-build kit form — or completely assembled and factory tested. All are supported by the best documentation in the business and by 62 service centers throughout the U.S. and Canada. You're never out in the cold.

For complete details and prices on Heathkit computers, peripherals and software, write today for the latest Heathkit Catalog or visit your nearby Heathkit Electronic Center*.

VISIT YOUR HEATHKIT STORE

Heathkit products are displayed, sold and serviced at Heathkit Electronic Centers*, located in major cities throughout the U.S. and Canada. See your white pages for the store nearest you.

*Units of Veritechnology Electronics Corporation, in the U.S.

Send for FREE catalog

Write to Heath Company, Dept. 351-734, Benton Harbor, MI 49022

In Canada, contact Heath Co., 1480 Dundas St. E., Mississauga, Ontario L4X 2R7

Heathkit®

236

IBM is a registered trademark of International Business Machines Corporation. CP/M is a trademark of Digital Research, Inc.

CP-196

The home computer you thought was years away is here.

C8P DF

Ohio Scientific's top of the line personal computer, the C8P DF. This system incorporates the most advanced technology now available in standard configurations and add-on options. The C8P DF has full capabilities as a personal computer, a small business computer, a home monitoring security system and an advanced process controller.

Personal Computer Features
The C8P DF features ultra-fast program execution. The standard model is twice as fast as other personal computers such as the Apple II and PET. The computer system is available with a GT option which nearly doubles the speed again, making it comparable to high end mini-computer systems. High speed execution makes elaborate video animation possible as well as other I/O functions which until now, have not been possible. The C8P DF features Ohio Scientific's 32 x 64 character display with graphics and gaming elements for an effective resolution of 256 x 512 points and up to 16 colors. Other features for personal use include a programmable tone generator from 200 to 20KHz and an 8 bit companding digital to analog converter for music and voice output, 2-8 axis joystick interfaces, and 2-10 key pad interfaces. Hundreds of personal applications, games and educational software packages are currently available for use with the C8P DF.

Business Applications
The C8P DF utilizes full size 8" floppy disks and is compatible with Ohio Scientific's advanced small business operating system, OS-65U and two types of information management systems, OS-MDMS and OS-DMS. The computer system comes standard with a high-speed printer interface and a modem interface. It features a full 53-key ASCII keyboard as well as 2048 character display with upper and lower case for business and word processing applications.

Home Control
The C8P DF has the most advanced home monitoring and control capabilities ever offered in a computer system. It incorporates a real time clock and a unique FOREGROUND/BACKGROUND operating system which allows the computer to function with normal BASIC programs at the same time it is monitoring external devices. The C8P DF comes standard with an AC remote control interface which allows it to control a wide range of AC appliances and lights remotely without wiring and an interface for home security systems which monitors fire, intrusion, car theft, water levels and freezer temperature, all without messy wiring. In addition, the C8P DF can accept Ohio Scientific's Votrax voice I/O board and/or Ohio Scientific's new universal telephone interface (UTI). The telephone interface connects the computer to any touch-tone or rotary dial telephone line. The computer system is able to answer calls, initiate calls and communicate via touch-tone signals, voice output or 300 baud modem signals. It can accept and decode touch-tone signals, 300 baud modem signals and record incoming voice messages. These features collectively give the C8P DF capabilities to monitor and control home functions with almost human-like capabilities.

Process Controller
The C8P DF incorporates a real time clock, FOREGROUND/BACKGROUND operation and 16 parallel I/O lines. Additionally a universal accessory BUS connector is accessible at the back of the computer to plug in additional 48 lines of parallel I/O and/or a complete analog signal I/O board with A/D and D/A and multiplexers.

Clearly, the C8P DF beats all existing small computers in conventional specifications plus it has capabilities far beyond any other computer system on the market today.

C8P DF is an 8-slot mainframe class computer with 32K static RAM, dual 8" floppies, and several open slots for expansion.

Prices start at under $3,000.

Computers come with keyboards and floppies where specified. Other equipment shown is optional.

For literature and the name of your local dealer, CALL 1-800-321-6850 TOLL FREE.

OHIO SCIENTIFIC
1333 SOUTH CHILLICOTHE ROAD
AURORA, OH 44202 • (216) 831-5600